PCI Express®
Base Specification
Revision 3.1a

December 7, 2015



Revision	Revision History	DATE
1.0	Initial release.	07/22/2002
1.0a	Incorporated Errata C1-C66 and E1-E4.17.	04/15/2003
1.1	Incorporated approved Errata and ECNs.	03/28/2005
2.0	Added 5.0 GT/s data rate and incorporated approved Errata and ECNs.	12/20/2006
2.1	Incorporated *Errata for the PCI Express Base Specification, Rev. 2.0* (February 27, 2009), and added the following ECNs: • Internal Error Reporting ECN (April 24, 2008) • Multicast ECN (December 14, 2007, approved by PWG May 8, 2008) • Atomic Operations ECN (January 15, 2008, approved by PWG April 17, 2008) • Resizable BAR Capability ECN (January 22, 2008, updated and approved by PWG April 24, 2008) • Dynamic Power Allocation ECN (May 24, 2008) • ID-Based Ordering ECN (January 16, 2008, updated 29 May 2008) • Latency Tolerance Reporting ECN (22 January 2008, updated 14 August 2008) • Alternative Routing-ID Interpretation (ARI) ECN (August 7, 2006, last updated June 4, 2007) • Extended Tag Enable Default ECN (September 5, 2008) • TLP Processing Hints ECN (September 11, 2008) • TLP Prefix ECN (December 15, 2008)	03/04/2009
3.0	Added 8.0 GT/s data rate, latest approved Errata, and the following ECNs: • Optimized Buffer Flush/Fill ECN (8 February 2008, updated 30 April 2009) • ASPM Optionality ECN (June 19, 2009, approved by the PWG August 20, 2009) • Incorporated End-End TLP Changes for RCs ECN (26 May 2010) and Protocol Multiplexing ECN (17 June 2010)	11/10/2010
3.1	Incorporated feedback from Member Review Incorporated Errata for the PCI Express® Base Specification Revision 3.0 ~~(November 7, 2013)~~ Incorporated MCPI-E Errata (3p1_active_errata_list_mpcie_28Aug2014.doc and 3p1_active_errata_list_mpcie_part2_11Sept2014.doc) Incorporated the following ECNs: • ECN: Downstream Port containment (DPC) • ECN: Separate Refclk Independent SSC (SRIS) Architecture • ECN: Process Address Space ID (PASID) • ECN: Lightweight Notification (LN) Protocol • ECN: Precision Time Measurement • ECN: Enhanced DPC (eDPC) • ECN: 8.0 GT/s Receiver Impedance • ECN: L1 PM Substates with CLKREQ • ECN: Change Root Complex Event Collector Class Code • ECN: M-PCIe • ~~RC~~ ECN: Readiness Notifications (RN) • ECN: Separate Refclk Independent SSC Architecture (SRIS) JTOL and SSC Profile Requirements	10/8/2014
3.1a	Minor update: Corrected: Equation 4.3.9 in Section 4.3.8.5., Separate Refclk With Independent SSC (SRIS) Architecture. Added missing square (exponent=2) in the definition of B. B = 2.2 × 10^12 × (2.π)^2 where ^= exponent.	12/5/2015

PCI-SIG® disclaims all warranties and liability for the use of this document and the information contained herein and assumes no responsibility for any errors that may appear in this document, nor does PCI-SIG make a commitment to update the information contained herein.

Contact the PCI-SIG office to obtain the latest revision of this specification.

Questions regarding the PCI Express Base Specification or membership in PCI-SIG may be forwarded to:

Membership Services
www.pcisig.com
E-mail: administration@pcisig.com
Phone: 503-619-0569
Fax: 503-644-6708

Technical Support
techsupp@pcisig.com

DISCLAIMER

This PCI Express Base Specification is provided "as is" with no warranties whatsoever, including any warranty of merchantability, noninfringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. PCI-SIG disclaims all liability for infringement of proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

PCI, PCI Express, PCIe, and PCI-SIG are trademarks or registered trademarks of PCI-SIG.
All other product names are trademarks, registered trademarks, or servicemarks of their respective owners.

Copyright © 2002-20154~~3~~ PCI-SIG

Contents

OBJECTIVE OF THE SPECIFICATION 35
DOCUMENT ORGANIZATION 35
DOCUMENTATION CONVENTIONS 35
TERMS AND ACRONYMS 36
REFERENCE DOCUMENTS 44
1. INTRODUCTION 46
1.1. A THIRD GENERATION I/O INTERCONNECT 46
1.2. PCI EXPRESS LINK 49
1.3. PCI EXPRESS FABRIC TOPOLOGY 50
1.3.1. Root Complex 50
1.3.2. Endpoints 51
1.3.3. Switch 54
1.3.4. Root Complex Event Collector 55
1.3.5. PCI Express to PCI/PCI-X Bridge 55
1.4. PCI EXPRESS FABRIC TOPOLOGY CONFIGURATION 55
1.5. PCI EXPRESS LAYERING OVERVIEW 56
1.5.1. Transaction Layer 57
1.5.2. Data Link Layer 57
1.5.3. Physical Layer 58
1.5.4. Layer Functions and Services 58
2. TRANSACTION LAYER SPECIFICATION 62
2.1. TRANSACTION LAYER OVERVIEW 62
2.1.1. Address Spaces, Transaction Types, and Usage 63
2.1.2. Packet Format Overview 65
2.2. TRANSACTION LAYER PROTOCOL - PACKET DEFINITION 67
2.2.1. Common Packet Header Fields 67
2.2.2. TLPs with Data Payloads - Rules 70
2.2.3. TLP Digest Rules 74
2.2.4. Routing and Addressing Rules 74
2.2.5. First/Last DW Byte Enables Rules 78
2.2.6. Transaction Descriptor 81
2.2.7. Memory, I/O, and Configuration Request Rules 87
2.2.8. Message Request Rules 94
2.2.9. Completion Rules 115
2.2.10. TLP Prefix Rules 118
2.3. HANDLING OF RECEIVED TLPS 123
2.3.1. Request Handling Rules 126
2.3.2. Completion Handling Rules 138
2.4. TRANSACTION ORDERING 142
2.4.1. Transaction Ordering Rules 142

2.4.2. Update Ordering and Granularity Observed by a Read Transaction 145
2.4.3. Update Ordering and Granularity Provided by a Write Transaction 146
2.5. VIRTUAL CHANNEL (VC) MECHANISM 147
2.5.1. Virtual Channel Identification (VC ID) 149
2.5.2. TC to VC Mapping 150
2.5.3. VC and TC Rules 151
2.6. ORDERING AND RECEIVE BUFFER FLOW CONTROL 152
2.6.1. Flow Control Rules 153
2.7. DATA INTEGRITY 164
2.7.1. ECRC Rules 164
2.7.2. Error Forwarding 168
2.8. COMPLETION TIMEOUT MECHANISM 170
2.9. LINK STATUS DEPENDENCIES 171
2.9.1. Transaction Layer Behavior in DL_Down Status 171
2.9.2. Transaction Layer Behavior in DL_Up Status 172
2.9.3. Transaction Layer Behavior During Downstream Port Containment 173

3. DATA LINK LAYER SPECIFICATION 175

3.1. DATA LINK LAYER OVERVIEW 175
3.2. DATA LINK CONTROL AND MANAGEMENT STATE MACHINE 177
3.2.1. Data Link Control and Management State Machine Rules 178
3.3. FLOW CONTROL INITIALIZATION PROTOCOL 180
3.3.1. Flow Control Initialization State Machine Rules 181
3.4. DATA LINK LAYER PACKETS (DLLPS) 184
3.4.1. Data Link Layer Packet Rules 184
3.5. DATA INTEGRITY 189
3.5.1. Introduction 189
3.5.2. LCRC, Sequence Number, and Retry Management (TLP Transmitter) 189
3.5.3. LCRC and Sequence Number (TLP Receiver) 202

4. PHYSICAL LAYER SPECIFICATION 211

4.1. INTRODUCTION 211
4.2. LOGICAL SUB-BLOCK 211
4.2.1. Encoding for 2.5 GT/s and 5.0 GT/s Data Rates 212
4.2.2. Encoding for 8.0 GT/s and Higher Data Rates 220
4.2.3. Link Equalization Procedure for 8.0 GT/s Data Rate 238
4.2.4. Link Initialization and Training 245
4.2.5. Link Training and Status State Machine (LTSSM) Descriptions 264
4.2.6. Link Training and Status State Rules 267
4.2.7. Clock Tolerance Compensation 332
4.2.8. Compliance Pattern in 8b/10b Encoding 337
4.2.9. Modified Compliance Pattern in 8b/10b Encoding 338
4.2.10. Compliance Pattern in 128b/130b Encoding 340
4.2.11. Modified Compliance Pattern in 128b/130b Encoding 342
4.3. ELECTRICAL SUB-BLOCK 343
4.3.1. Electrical Specification Organization 343
4.3.2. Interoperability Criteria for 2.5, 5.0, and 8.0 GT/s Devices 343

4.3.3. Transmitter Specification 345
4.3.4. Receiver Specifications 378
4.3.5. Low Frequency and Miscellaneous Signaling Requirements 404
4.3.6. Channel Specification 408
4.3.7. Refclk Specifications 422
4.3.8. Refclk Specifications for 8.0 GT/s 433

5. POWER MANAGEMENT 441

5.1. OVERVIEW 441
5.1.1. Statement of Requirements 442
5.2. LINK STATE POWER MANAGEMENT 442
5.3. PCI-PM SOFTWARE COMPATIBLE MECHANISMS 448
5.3.1. Device Power Management States (D-States) of a Function 448
5.3.2. PM Software Control of the Link Power Management State 452
5.3.3. Power Management Event Mechanisms 457
5.4. NATIVE PCI EXPRESS POWER MANAGEMENT MECHANISMS 464
5.4.1. Active State Power Management (ASPM) 464
5.5. L1 PM SUBSTATES 484
5.5.1. Entry conditions for L1 PM Substates and L1.0 Requirements 488
5.5.2. L1.1 Requirements 489
5.5.3. L1.2 Requirements 490
5.5.4. L1 PM Substates Configuration 494
5.5.5. L1 PM Substates Timing Parameters 496
5.6. AUXILIARY POWER SUPPORT 496
5.6.1. Auxiliary Power Enabling 496
5.7. POWER MANAGEMENT SYSTEM MESSAGES AND DLLPS 498

6. SYSTEM ARCHITECTURE 499

6.1. INTERRUPT AND PME SUPPORT 499
6.1.1. Rationale for PCI Express Interrupt Model 499
6.1.2. PCI Compatible INTx Emulation 500
6.1.3. INTx Emulation Software Model 500
6.1.4. Message Signaled Interrupt (MSI/MSI-X) Support 500
6.1.5. PME Support 501
6.1.6. Native PME Software Model 502
6.1.7. Legacy PME Software Model 502
6.1.8. Operating System Power Management Notification 503
6.1.9. PME Routing Between PCI Express and PCI Hierarchies 503
6.2. ERROR SIGNALING AND LOGGING 503
6.2.1. Scope 504
6.2.2. Error Classification 504
6.2.3. Error Signaling 506
6.2.4. Error Logging 514
6.2.5. Sequence of Device Error Signaling and Logging Operations 520
6.2.6. Error Message Controls 522
6.2.7. Error Listing and Rules 523
6.2.8. Virtual PCI Bridge Error Handling 528

6.2.9. Internal Errors 530
6.2.10. Downstream Port Containment (DPC) 530
6.3. VIRTUAL CHANNEL SUPPORT 541
6.3.1. Introduction and Scope 541
6.3.2. TC/VC Mapping and Example Usage 541
6.3.3. VC Arbitration 543
6.3.4. Isochronous Support 551
6.4. DEVICE SYNCHRONIZATION 554
6.5. LOCKED TRANSACTIONS 555
6.5.1. Introduction 555
6.5.2. Initiation and Propagation of Locked Transactions - Rules 556
6.5.3. Switches and Lock - Rules 557
6.5.4. PCI Express/PCI Bridges and Lock - Rules 557
6.5.5. Root Complex and Lock - Rules 558
6.5.6. Legacy Endpoints 558
6.5.7. PCI Express Endpoints 558
6.6. PCI EXPRESS RESET - RULES 558
6.6.1. Conventional Reset 558
6.6.2. Function-Level Reset (FLR) 561
6.7. PCI EXPRESS HOT-PLUG SUPPORT 566
6.7.1. Elements of Hot-Plug 566
6.7.2. Registers Grouped by Hot-Plug Element Association 572
6.7.3. PCI Express Hot-Plug Events 574
6.7.4. Firmware Support for Hot-Plug 577
6.7.5. Async Removal 577
6.8. POWER BUDGETING CAPABILITY 579
6.8.1. System Power Budgeting Process Recommendations 579
6.9. SLOT POWER LIMIT CONTROL 580
6.10. ROOT COMPLEX TOPOLOGY DISCOVERY 583
6.11. LINK SPEED MANAGEMENT 585
6.12. ACCESS CONTROL SERVICES (ACS) 586
6.12.1. ACS Component Capability Requirements 587
6.12.2. Interoperability 591
6.12.3. ACS Peer-to-Peer Control Interactions 592
6.12.4. ACS Violation Error Handling 593
6.12.5. ACS Redirection Impacts on Ordering Rules 593
6.13. ALTERNATIVE ROUTING-ID INTERPRETATION (ARI) 596
6.14. MULTICAST OPERATIONS 600
6.14.1. Multicast TLP Processing 600
6.14.2. Multicast Ordering 603
6.14.3. Multicast Capability Structure Field Updates 603
6.14.4. MC Blocked TLP Processing 604
6.14.5. MC_Overlay Mechanism 604
6.15. ATOMIC OPERATIONS (ATOMICOPS) 608
6.15.1. AtomicOp Use Models and Benefits 609
6.15.2. AtomicOp Transaction Protocol Summary 609

6.15.3. Root Complex Support for AtomicOps 611
6.15.4. Switch Support for AtomicOps 612
6.16. DYNAMIC POWER ALLOCATION (DPA) CAPABILITY 613
6.16.1. DPA Capability with Multi-Function Devices 614
6.17. TLP PROCESSING HINTS (TPH) 614
6.17.1. Processing Hints 614
6.17.2. Steering Tags 615
6.17.3. ST Modes of Operation 616
6.17.4. TPH Capability 617
6.18. LATENCY TOLERANCE REPORTING (LTR) MECHANISM 618
6.19. OPTIMIZED BUFFER FLUSH/FILL (OBFF) MECHANISM 624
6.20. PASID TLP PREFIX 628
6.20.1. Managing PASID TLP Prefix Usage 628
6.20.2. PASID TLP Layout 629
6.21. LIGHTWEIGHT NOTIFICATION (LN) PROTOCOL 633
6.21.1. LN Protocol Operation 634
6.21.2. LN Registration Management 635
6.21.3. LN Ordering Considerations 636
6.21.4. LN Software Configuration 636
6.21.5. LN Protocol Summary 637
6.22. PRECISION TIME MEASUREMENT (PTM) MECHANISM 639
6.22.1. Introduction 639
6.22.2. PTM Link Protocol 641
6.22.3. Configuration and Operational Requirements 644
6.23. READINESS NOTIFICATIONS (RN) 650
6.23.1. Device Readiness Status (DRS) 650
6.23.2. Function Readiness Status (FRS) 651
6.23.3. FRS Queuing 652

7. SOFTWARE INITIALIZATION AND CONFIGURATION 653

7.1. CONFIGURATION TOPOLOGY 653
7.2. PCI EXPRESS CONFIGURATION MECHANISMS 654
7.2.1. PCI 3.0 Compatible Configuration Mechanism 655
7.2.2. PCI Express Enhanced Configuration Access Mechanism (ECAM) 656
7.2.3. Root Complex Register Block 660
7.3. CONFIGURATION TRANSACTION RULES 661
7.3.1. Device Number 661
7.3.2. Configuration Transaction Addressing 662
7.3.3. Configuration Request Routing Rules 662
7.3.4. PCI Special Cycles 663
7.4. CONFIGURATION REGISTER TYPES 664
7.5. PCI-COMPATIBLE CONFIGURATION REGISTERS 665
7.5.1. Type 0/1 Common Configuration Space 665
7.5.2. Type 0 Configuration Space Header 673
7.5.3. Type 1 Configuration Space Header 675
7.6. PCI POWER MANAGEMENT CAPABILITY STRUCTURE 679
7.7. MSI AND MSI-X CAPABILITY STRUCTURES 681

7.7.1. Vector Control for MSI-X Table Entries 681
7.8. PCI EXPRESS CAPABILITY STRUCTURE 682
7.8.1. PCI Express Capability List Register (Offset 00h) 683
7.8.2. PCI Express Capabilities Register (Offset 02h) 684
7.8.3. Device Capabilities Register (Offset 04h) 686
7.8.4. Device Control Register (Offset 08h) 691
7.8.5. Device Status Register (Offset 0Ah) 698
7.8.6. Link Capabilities Register (Offset 0Ch) 700
7.8.7. Link Control Register (Offset 10h) 705
7.8.8. Link Status Register (Offset 12h) 713
7.8.9. Slot Capabilities Register (Offset 14h) 717
7.8.10. Slot Control Register (Offset 18h) 719
7.8.11. Slot Status Register (Offset 1Ah) 723
7.8.12. Root Control Register (Offset 1Ch) 725
7.8.13. Root Capabilities Register (Offset 1Eh) 726
7.8.14. Root Status Register (Offset 20h) 727
7.8.15. Device Capabilities 2 Register (Offset 24h) 728
7.8.16. Device Control 2 Register (Offset 28h) 734
7.8.17. Device Status 2 Register (Offset 2Ah) 738
7.8.18. Link Capabilities 2 Register (Offset 2Ch) 738
7.8.19. Link Control 2 Register (Offset 30h) 741
7.8.20. Link Status 2 Register (Offset 32h) 746
7.8.21. Slot Capabilities 2 Register (Offset 34h) 750
7.8.22. Slot Control 2 Register (Offset 38h) 750
7.8.23. Slot Status 2 Register (Offset 3Ah) 750
7.9. PCI EXPRESS EXTENDED CAPABILITIES 750
7.9.1. Extended Capabilities in Configuration Space 751
7.9.2. Extended Capabilities in the Root Complex Register Block 751
7.9.3. PCI Express Extended Capability Header 751
7.10. ADVANCED ERROR REPORTING CAPABILITY 752
7.10.1. Advanced Error Reporting Extended Capability Header (Offset 00h) 753
7.10.2. Uncorrectable Error Status Register (Offset 04h) 754
7.10.3. Uncorrectable Error Mask Register (Offset 08h) 756
7.10.4. Uncorrectable Error Severity Register (Offset 0Ch) 758
7.10.5. Correctable Error Status Register (Offset 10h) 760
7.10.6. Correctable Error Mask Register (Offset 14h) 761
7.10.7. Advanced Error Capabilities and Control Register (Offset 18h) 762
7.10.8. Header Log Register (Offset 1Ch) 764
7.10.9. Root Error Command Register (Offset 2Ch) 765
7.10.10. Root Error Status Register (Offset 30h) 766
7.10.11. Error Source Identification Register (Offset 34h) 769
7.10.12. TLP Prefix Log Register (Offset 38h) 769
7.11. VIRTUAL CHANNEL CAPABILITY 770
7.11.1. Virtual Channel Extended Capability Header (Offset 00h) 772
7.11.2. Port VC Capability Register 1 (Offset 04h) 772
7.11.3. Port VC Capability Register 2 (Offset 08h) 774

7.11.4. Port VC Control Register (Offset 0Ch) *775*
7.11.5. Port VC Status Register (Offset 0Eh) *776*
7.11.6. VC Resource Capability Register *777*
7.11.7. VC Resource Control Register *779*
7.11.8. VC Resource Status Register *781*
7.11.9. VC Arbitration Table *782*
7.11.10. Port Arbitration Table *783*
7.12. DEVICE SERIAL NUMBER CAPABILITY 785
7.12.1. Device Serial Number Extended Capability Header (Offset 00h) *785*
7.12.2. Serial Number Register (Offset 04h) *786*
7.13. PCI EXPRESS ROOT COMPLEX LINK DECLARATION CAPABILITY 787
7.13.1. Root Complex Link Declaration Extended Capability Header (Offset 00h) *789*
7.13.2. Element Self Description (Offset 04h) *790*
7.13.3. Link Entries *791*
7.14. PCI EXPRESS ROOT COMPLEX INTERNAL LINK CONTROL CAPABILITY 795
7.14.1. Root Complex Internal Link Control Extended Capability Header (Offset 00h) *795*
7.14.2. Root Complex Link Capabilities Register (Offset 04h) *796*
7.14.3. Root Complex Link Control Register (Offset 08h) *800*
7.14.4. Root Complex Link Status Register (Offset 0Ah) *801*
7.15. POWER BUDGETING CAPABILITY 802
7.15.1. Power Budgeting Extended Capability Header (Offset 00h) *803*
7.15.2. Data Select Register (Offset 04h) *803*
7.15.3. Data Register (Offset 08h) *804*
7.15.4. Power Budget Capability Register (Offset 0Ch) *806*
7.16. ACS EXTENDED CAPABILITY 807
7.16.1. ACS Extended Capability Header (Offset 00h) *807*
7.16.2. ACS Capability Register (Offset 04h) *808*
7.16.3. ACS Control Register (Offset 06h) *809*
7.16.4. Egress Control Vector (Offset 08h) *810*
7.17. PCI EXPRESS ROOT COMPLEX EVENT COLLECTOR ENDPOINT ASSOCIATION CAPABILITY 812
7.17.1. Root Complex Event Collector Endpoint Association Extended Capability Header (Offset 00h) *812*
7.17.2. Association Bitmap for Root Complex Integrated Endpoints (Offset 04h) *813*
7.18. MULTI-FUNCTION VIRTUAL CHANNEL CAPABILITY 814
7.18.1. MFVC Extended Capability Header (Offset 00h) *815*
7.18.2. Port VC Capability Register 1 (Offset 04h) *815*
7.18.3. Port VC Capability Register 2 (Offset 08h) *817*
7.18.4. Port VC Control Register (Offset 0Ch) *818*
7.18.5. Port VC Status Register (Offset 0Eh) *818*
7.18.6. VC Resource Capability Register *819*
7.18.7. VC Resource Control Register *821*
7.18.8. VC Resource Status Register *823*
7.18.9. VC Arbitration Table *824*
7.18.10. Function Arbitration Table *824*
7.19. VENDOR-SPECIFIC CAPABILITY 826

7.19.1. Vendor-Specific Extended Capability Header (Offset 00h) *827*
7.19.2. Vendor-Specific Header (Offset 04h) *828*
7.20. RCRB HEADER CAPABILITY 829
7.20.1. RCRB Header Extended Capability Header (Offset 00h) *829*
7.20.2. Vendor ID (Offset 04h) and Device ID (Offset 06h) *830*
7.20.3. RCRB Capabilities (Offset 08h) *831*
7.20.4. RCRB Control (Offset 0Ch) *831*
7.21. MULTICAST CAPABILITY 832
7.21.1. Multicast Extended Capability Header (Offset 00h) *832*
7.21.2. Multicast Capability Register (Offset 04h) *833*
7.21.3. Multicast Control Register (Offset 06h) *834*
7.21.4. MC_Base_Address Register (Offset 08h) *835*
7.21.5. MC_Receive Register (Offset 10h) *835*
7.21.6. MC_Block_All Register (Offset 18h) *836*
7.21.7. MC_Block_Untranslated Register (Offset 20h) *836*
7.21.8. MC_Overlay_BAR (Offset 28h) *837*
7.22. RESIZABLE BAR CAPABILITY 838
7.22.1. Resizable BAR Extended Capability Header (Offset 00h) *840*
7.22.2. Resizable BAR Capability Register *840*
7.22.3. Resizable BAR Control Register *842*
7.23. ARI CAPABILITY 843
7.23.1. ARI Capability Header (Offset 00h) *844*
7.23.2. ARI Capability Register (Offset 04h) *844*
7.23.3. ARI Control Register (Offset 06h) *845*
7.24. DYNAMIC POWER ALLOCATION (DPA) CAPABILITY 846
7.24.1. DPA Extended Capability Header (Offset 00h) *846*
7.24.2. DPA Capability Register (Offset 04h) *847*
7.24.3. DPA Latency Indicator Register (Offset 08h) *848*
7.24.4. DPA Status Register (Offset 0Ch) *848*
7.24.5. DPA Control Register (Offset 0Eh) *849*
7.24.6. DPA Power Allocation Array *849*
7.25. LATENCY TOLERANCE REPORTING (LTR) CAPABILITY 850
7.25.1. LTR Extended Capability Header (Offset 00h) *850*
7.25.2. Max Snoop Latency Register (Offset 04h) *851*
7.25.3. Max No-Snoop Latency Register (Offset 06h) *851*
7.26. TPH REQUESTER CAPABILITY 852
7.26.1. TPH Requester Extended Capability Header (Offset 00h) *853*
7.26.2. TPH Requester Capability Register (Offset 04h) *853*
7.26.3. TPH Requester Control Register (Offset 08h) *855*
7.26.4. TPH ST Table (Starting from Offset 0Ch) *856*
7.27. SECONDARY PCI EXPRESS EXTENDED CAPABILITY 857
7.27.1. Secondary PCI Express Extended Capability Header (Offset 00h) *857*
7.27.2. Link Control 3 Register (Offset 04h) *858*
7.27.3. Lane Error Status Register (Offset 08h) *859*
7.27.4. Lane Equalization Control Register (Offset 0Ch) *860*
7.28. M-PCIE EXTENDED CAPABILITY 864

7.28.1. M-PCIe Extended Capability Header (Offset 00h) ... 864
7.28.2. M-PCIe Capabilities Register (Offset 04h) ... 866
7.28.3. M-PCIe Control Register (Offset 08h) ... 867
7.28.4. M-PCIe Status Register (Offset 0Ch) ... 868
7.28.5. M-PCIe LANE Error Status Register (Offset 10h) ... 869
7.28.6. M-PCIe Phy Control Address Register (Offset 14h) ... 870
7.28.7. M-PCIe Phy Control Data Register (Offset 18h) ... 871
7.29. PASID EXTENDED CAPABILITY STRUCTURE ... 872
7.29.1. PASID Extended Capability Header (Offset 00h) ... 873
7.29.2. PASID Capability Register (Offset 04h) ... 874
7.29.3. PASID Control Register (Offset 06h) ... 875
7.30. LNR EXTENDED CAPABILITY ... 876
7.30.1. LNR Extended Capability Header (Offset 00h) ... 876
7.30.2. LNR Capability Register (Offset 04h) ... 877
7.30.3. LNR Control Register (Offset 04h) ... 878
7.31. DPC EXTENDED CAPABILITY ... 878
7.31.1. DPC Extended Capability Header (Offset 00h) ... 880
7.31.2. DPC Capability Register (Offset 04h) ... 881
7.31.3. DPC Control Register (Offset 06h) ... 883
7.31.4. DPC Status Register (Offset 08h) ... 885
7.31.5. DPC Error Source ID Register (Offset 0Ah) ... 887
7.31.6. RP PIO Status Register (Offset 0Ch) ... 888
7.31.7. RP PIO Mask Register (Offset 10h) ... 889
7.31.8. RP PIO Severity Register (Offset 14h) ... 890
7.31.9. RP PIO SysError Register (Offset 18h) ... 891
7.31.10. RP PIO Exception Register (Offset 1Ch) ... 892
7.31.11. RP PIO Header Log Register (Offset 20h) ... 893
7.31.12. RP PIO ImpSpec Log Register (Offset 30h) ... 893
7.31.13. RP PIO TLP Prefix Log Register (Offset 34h) ... 894
7.32. PRECISION TIME MANAGEMENT (PTM) CAPABILITY ... 895
7.32.1. PTM Extended Capability Header (Offset 00h) ... 896
7.32.2. PTM Capability Register (Offset 04h) ... 897
7.32.3. PTM Control Register (Offset 08h) ... 898
7.33. L1 PM SUBSTATES EXTENDED CAPABILITY ... 900
7.33.1. L1 PM Substates Extended Capability Header (Offset 00h) ... 900
7.33.2. L1 PM Substates Capabilities Register (Offset 04h) ... 901
7.33.3. L1 PM Substates Control 1 Register (Offset 08h) ... 903
7.33.4. L1 PM Substates Control 2 Register (Offset 0Ch) ... 905
7.34. FUNCTION READINESS STATUS (FRS) QUEUING EXTENDED CAPABILITY ... 906
7.34.1. Function Readiness Status (FRS) Queuing Extended Capability Header (Offset 00h) 907
7.34.2. FRS Queuing Capability Register (Offset 04h) ... 908
7.34.3. FRS Queuing Status Register (Offset 08h) ... 909
7.34.4. FRS Queuing Control Register (Offset 0Ah) ... 909
7.34.5. FRS Message Queue Register (Offset 0Ch) ... 910
7.35. READINESS TIME REPORTING EXTENDED CAPABILITY ... 911

7.35.1. Readiness Time Reporting Extended Capability Header (Offset 00h) 913
7.35.2. Readiness Time Reporting 1 (Offset 04h) 914
7.35.3. Readiness Time Reporting 2 (Offset 08h) 915

8. M-PCIE LOGICAL SUB-BLOCK 916

8.1. PHY REQUIREMENTS 918
8.2. CONFIGURATION 919
8.2.1. Link Discovery and Configuration 919
8.2.2. Attributes 920
8.2.3. Remote Register Access Protocol (RRAP): 933
8.3. SYMBOL ENCODING, FRAMING AND SCRAMBLING 946
8.3.1. 8b/10b Decode Rules 946
8.3.2. Framing and Application of Symbols to LANES 946
8.3.3. Data Scrambling 947
8.4. LINK INITIALIZATION AND TRAINING 947
8.4.1. Training Sequence (TS) Ordered Sets 948
8.4.2. Electrical Idle 954
8.4.3. EIEOS for M-PCIe 954
8.4.4. Lane Polarity Inversion 954
8.4.5. Fast Training Sequence (FTS) 954
8.4.6. LINK Data RATE 955
8.4.7. LINK Width 955
8.4.8. LANE-to-LANE De-skew 955
8.4.9. LINK Training and Status State Machine (LTSSM) 955
8.4.10. Entry to HIBERN8 977
8.5. RECEIVER ERROR 977
8.6. CLOCK TOLERANCE COMPENSATION 978
8.7. DYNAMIC LINK BANDWIDTH MANAGEMENT 978
8.7.1. LINK Rate Series and Speed Management 978
8.7.2. LINK Width Management 979
8.7.3. Dynamic LINK Re-Configuration 979
8.8. M-PHY REGISTERS 982
8.8.1. M-PHY Capability Registers 982
8.8.2. M-PHY Configuration Attributes 990

A. ISOCHRONOUS APPLICATIONS **993**

A.1. INTRODUCTION 993
A.2. ISOCHRONOUS CONTRACT AND CONTRACT PARAMETERS 995
A.2.1. Isochronous Time Period and Isochronous Virtual Timeslot 996
A.2.2. Isochronous Payload Size 997
A.2.3. Isochronous Bandwidth Allocation 997
A.2.4. Isochronous Transaction Latency 998
A.2.5. An Example Illustrating Isochronous Parameters 999
A.3. ISOCHRONOUS TRANSACTION RULES 1000
A.4. TRANSACTION ORDERING 1000
A.5. ISOCHRONOUS DATA COHERENCY 1000
A.6. FLOW CONTROL 1001

A.7. CONSIDERATIONS FOR BANDWIDTH ALLOCATION 1001
A.7.1. Isochronous Bandwidth of PCI Express Links 1001
A.7.2. Isochronous Bandwidth of Endpoints 1001
A.7.3. Isochronous Bandwidth of Switches 1001
A.7.4. Isochronous Bandwidth of Root Complex 1002
A.8. CONSIDERATIONS FOR PCI EXPRESS COMPONENTS 1002
A.8.1. An Endpoint as a Requester 1002
A.8.2. An Endpoint as a Completer 1002
A.8.3. Switches 1003
A.8.4. Root Complex 1004

B. SYMBOL ENCODING 1005

C. PHYSICAL LAYER APPENDIX 1014

C.1. 8B/10B DATA SCRAMBLING EXAMPLE 1014
C.2. 128B/130B DATA SCRAMBLING EXAMPLE 1020

D. REQUEST DEPENDENCIES 1023

E. ID-BASED ORDERING USAGE 1026

E.1. INTRODUCTION 1026
E.2. POTENTIAL BENEFITS WITH IDO USE 1027
E.2.1. Benefits for MFD/RP Direct Connect 1027
E.2.2. Benefits for Switched Environments 1027
E.2.3. Benefits for Integrated Endpoints 1028
E.2.4. IDO Use in Conjunction with RO 1028
E.3. WHEN TO USE IDO 1028
E.4. WHEN NOT TO USE IDO 1029
E.4.1. When Not to Use IDO with Endpoints 1029
E.4.2. When Not to Use IDO with Root Ports 1029
E.5. SOFTWARE CONTROL OF IDO USE 1030
E.5.1. Software Control of Endpoint IDO Use 1030
E.5.2. Software Control of Root Port IDO Use 1031

F. MESSAGE CODE USAGE 1032

G. PROTOCOL MULTIPLEXING 1034

G.1. PROTOCOL MULTIPLEXING INTERACTIONS WITH PCI EXPRESS 1037
G.2. PMUX PACKETS 1043
G.3. PMUX PACKET LAYOUT 1044
G.3.1. PMUX Packet Layout for 8b10b Encoding 1044
G.3.2. PMUX Packet Layout at 128b/130b Encoding 1046
G.4. PMUX CONTROL 1049
G.5. PMUX EXTENDED CAPABILITY 1049
G.5.1. PCI Express Extended Header (Offset 00h) 1050
G.5.2. PMUX Capability Register (Offset 04h) 1051
G.5.3. PMUX Control Register (Offset 08h) 1052
G.5.4. PMUX Status Register (Offset 0Ch) 1054
G.5.5. PMUX Protocol Array (Offsets 10h Through 48h) 1057

H. M-PCIE TIMING DIAGRAMS 1059

H.1. INIT TO L0 1060
H.2. L0 WITH TRANSMITTER IN STALL 1061
H.3. L0 TO L1 1062
H.4. DOWNSTREAM PORT INITIATED LINK BANDWIDTH CHANGE 1063
H.5. UPSTREAM PORT INITIATED LINK BANDWIDTH CHANGE 1064

I. M-PCIE COMPLIANCE PATTERNS 1065

I.1. RPAT 1065
I.2. RPAT VARIATION BY LANE 1066
I.3. CONTINUOUS MODE CRPAT 1066
I.4. BURST MODE CRPAT 1067

ACKNOWLEDGEMENTS **1068**

Figures

FIGURE 1-1: PCI EXPRESS LINK 49
FIGURE 1-2: EXAMPLE TOPOLOGY 50
FIGURE 1-3: LOGICAL BLOCK DIAGRAM OF A SWITCH 54
FIGURE 1-4: HIGH-LEVEL LAYERING DIAGRAM 56
FIGURE 1-5: PACKET FLOW THROUGH THE LAYERS 57
FIGURE 2-1: LAYERING DIAGRAM HIGHLIGHTING THE TRANSACTION LAYER 62
FIGURE 2-2: SERIAL VIEW OF A TLP 65
FIGURE 2-3: GENERIC TLP FORMAT 66
FIGURE 2-4: FIELDS PRESENT IN ALL TLPS 67
FIGURE 2-5: FIELDS PRESENT IN ALL TLP HEADERS 68
FIGURE 2-6: EXAMPLES OF COMPLETER TARGET MEMORY ACCESS FOR FETCHADD 73
FIGURE 2-7: 64-BIT ADDRESS ROUTING 75
FIGURE 2-8: 32-BIT ADDRESS ROUTING 75
FIGURE 2-9: ID ROUTING WITH 4 DW HEADER 77
FIGURE 2-10: ID ROUTING WITH 3 DW HEADER 78
FIGURE 2-11: LOCATION OF BYTE ENABLES IN TLP HEADER 79
FIGURE 2-12: TRANSACTION DESCRIPTOR 81
FIGURE 2-13: TRANSACTION ID 82
FIGURE 2-14: ATTRIBUTES FIELD OF TRANSACTION DESCRIPTOR 84
FIGURE 2-15: REQUEST HEADER FORMAT FOR 64-BIT ADDRESSING OF MEMORY 88
FIGURE 2-16: REQUEST HEADER FORMAT FOR 32-BIT ADDRESSING OF MEMORY 88
FIGURE 2-17: REQUEST HEADER FORMAT FOR I/O TRANSACTIONS 89
FIGURE 2-18: REQUEST HEADER FORMAT FOR CONFIGURATION TRANSACTIONS 90
FIGURE 2-19: TPH TLP PREFIX 91
FIGURE 2-20: LOCATION OF PH[1:0] IN A 4 DW REQUEST HEADER 91
FIGURE 2-21: LOCATION OF PH[1:0] IN A 3 DW REQUEST HEADER 92
FIGURE 2-22: LOCATION OF ST[7:0] IN THE MEMORY WRITE REQUEST HEADER 93
FIGURE 2-23: LOCATION OF ST[7:0] IN MEMORY READ AND ATOMICOP REQUEST HEADERS 93
FIGURE 2-24: MESSAGE REQUEST HEADER 95
FIGURE 2-25: HEADER FOR VENDOR-DEFINED MESSAGES 105
FIGURE 2-26: HEADER FOR PCI-SIG-DEFINED VDMS 106
FIGURE 2-27: LN MESSAGE 108
FIGURE 2-28: DRS MESSAGE 109
FIGURE 2-29: FRS MESSAGE 111
FIGURE 2-30: LTR MESSAGE 112
FIGURE 2-31: OBFF MESSAGE 113
FIGURE 2-32: PTM REQUEST/RESPONSE MESSAGE 114
FIGURE 2-33: PTM RESPONSED MESSAGE (4 DW HEADER AND 1 DW PAYLOAD) 115
FIGURE 2-34: COMPLETION HEADER FORMAT 116
FIGURE 2-35: (NON-ARI) COMPLETER ID 117
FIGURE 2-36: ARI COMPLETER ID 117
FIGURE 2-37: FLOWCHART FOR HANDLING OF RECEIVED TLPS 124
FIGURE 2-38: FLOWCHART FOR SWITCH HANDLING OF TLPS 126

FIGURE 2-39: FLOWCHART FOR HANDLING OF RECEIVED REQUEST 131
FIGURE 2-40: VIRTUAL CHANNEL CONCEPT – AN ILLUSTRATION 148
FIGURE 2-41: VIRTUAL CHANNEL CONCEPT – SWITCH INTERNALS (UPSTREAM FLOW) 148
FIGURE 2-42: AN EXAMPLE OF TC/VC CONFIGURATIONS 151
FIGURE 2-43: RELATIONSHIP BETWEEN REQUESTER AND ULTIMATE COMPLETER 152
FIGURE 2-44: CALCULATION OF 32-BIT ECRC FOR TLP END TO END DATA INTEGRITY PROTECTION 167
FIGURE 3-1: LAYERING DIAGRAM HIGHLIGHTING THE DATA LINK LAYER 175
FIGURE 3-2: DATA LINK CONTROL AND MANAGEMENT STATE MACHINE 178
FIGURE 3-3: VC0 FLOW CONTROL INITIALIZATION EXAMPLE WITH 8B/10B ENCODING-BASED FRAMING 183
FIGURE 3-4: DLLP TYPE AND CRC FIELDS 184
FIGURE 3-5: DATA LINK LAYER PACKET FORMAT FOR ACK AND NAK 186
FIGURE 3-6: DATA LINK LAYER PACKET FORMAT FOR INITFC1 186
FIGURE 3-7: DATA LINK LAYER PACKET FORMAT FOR INITFC2 186
FIGURE 3-8: DATA LINK LAYER PACKET FORMAT FOR UPDATEFC 187
FIGURE 3-9: PM DATA LINK LAYER PACKET FORMAT 187
FIGURE 3-10: VENDOR SPECIFIC DATA LINK LAYER PACKET FORMAT 187
FIGURE 3-11: DIAGRAM OF CRC CALCULATION FOR DLLPS 188
FIGURE 3-12: TLP WITH LCRC AND TLP SEQUENCE NUMBER APPLIED 189
FIGURE 3-13: TLP FOLLOWING APPLICATION OF TLP SEQUENCE NUMBER AND RESERVED BITS 191
FIGURE 3-14: CALCULATION OF LCRC 193
FIGURE 3-15: RECEIVED DLLP ERROR CHECK FLOWCHART 201
FIGURE 3-16: ACK/NAK DLLP PROCESSING FLOWCHART 202
FIGURE 3-17: RECEIVE DATA LINK LAYER HANDLING OF TLPS 206
FIGURE 4-1: LAYERING DIAGRAM HIGHLIGHTING PHYSICAL LAYER 211
FIGURE 4-2: CHARACTER TO SYMBOL MAPPING 212
FIGURE 4-3: BIT TRANSMISSION ORDER ON PHYSICAL LANES - X1 EXAMPLE 213
FIGURE 4-4: BIT TRANSMISSION ORDER ON PHYSICAL LANES - X4 EXAMPLE 213
FIGURE 4-5: TLP WITH FRAMING SYMBOLS APPLIED 216
FIGURE 4-6: DLLP WITH FRAMING SYMBOLS APPLIED 217
FIGURE 4-7: FRAMED TLP ON A X1 LINK 217
FIGURE 4-8: FRAMED TLP ON A X2 LINK 218
FIGURE 4-9: FRAMED TLP ON A X4 LINK 218
FIGURE 4-10: LFSR WITH SCRAMBLING POLYNOMIAL 220
FIGURE 4-11: EXAMPLE OF BIT TRANSMISSION ORDER IN A X1 LINK SHOWING 130 BITS OF A BLOCK 221
FIGURE 4-12: EXAMPLE OF BIT PLACEMENT IN A X4 LINK WITH ONE BLOCK PER LANE 221
FIGURE 4-13: LAYOUT OF FRAMING TOKENS 225
FIGURE 4-14: TLP AND DLLP LAYOUT 227
FIGURE 4-15: PACKET TRANSMISSION IN A X8 LINK 227
FIGURE 4-16: NULLIFIED TLP LAYOUT IN A X8 LINK WITH OTHER PACKETS 228
FIGURE 4-17: SKP ORDERED SET OF LENGTH 66-BIT IN A X8 LINK 228
FIGURE 4-18: LFSR WITH SCRAMBLING POLYNOMIAL IN 8.0 GT/S AND ABOVE DATA RATE 235
FIGURE 4-19: ALTERNATE IMPLEMENTATION OF THE LFSR FOR DESCRAMBLING 237

FIGURE 4-20: EQUALIZATION FLOW 243
FIGURE 4-21: ELECTRICAL IDLE EXIT ORDERED SET FOR 8.0 GT/S AND ABOVE DATA RATES 254
FIGURE 4-22: MAIN STATE DIAGRAM FOR LINK TRAINING AND STATUS STATE MACHINE 269
FIGURE 4-23: DETECT SUBSTATE MACHINE 271
FIGURE 4-24: POLLING SUBSTATE MACHINE 279
FIGURE 4-25: CONFIGURATION SUBSTATE MACHINE 294
FIGURE 4-26: RECOVERY SUBSTATE MACHINE 315
FIGURE 4-27: L0S SUBSTATE MACHINE 322
FIGURE 4-28: L1 SUBSTATE MACHINE 324
FIGURE 4-29: L2 SUBSTATE MACHINE 326
FIGURE 4-30: LOOPBACK SUBSTATE MACHINE 331
FIGURE 4-31: TRANSMITTER, CHANNEL, AND RECEIVER BOUNDARIES 345
FIGURE 4-32: REQUIRED SETUP FOR CHARACTERIZING A 5.0 GT/S TRANSMITTER 346
FIGURE 4-33: ALLOWABLE SETUP FOR CHARACTERIZING A 2.5 GT/S TRANSMITTER 346
FIGURE 4-34: TX TEST BOARD EXAMPLE 347
FIGURE 4-35: SINGLE-ENDED AND DIFFERENTIAL LEVELS 349
FIGURE 4-36: FULL SWING SIGNALING VOLTAGE PARAMETERS SHOWING -6 DB DE-EMPHASIS 350
FIGURE 4-37: REDUCED SWING TX PARAMETERS 350
FIGURE 4-38: MINIMUM PULSE WIDTH DEFINITION 351
FIGURE 4-39: FULL SWING TX PARAMETERS SHOWING DE-EMPHASIS 352
FIGURE 4-40: MEASURING FULL SWING/DE-EMPHASIZED VOLTAGES FROM EYE DIAGRAM 353
FIGURE 4-41: TX EQUALIZATION FIR REPRESENTATION 355
FIGURE 4-42: DEFINITION OF TX VOLTAGE LEVELS AND EQUALIZATION RATIOS 356
FIGURE 4-43: WAVEFORM MEASUREMENT POINTS FOR PRE-SHOOT AND DE-EMPHASIS 357
FIGURE 4-44: $V_{TX\text{-}FS\text{-}NO\text{-}EQ}$ MEASUREMENT 360
FIGURE 4-45: TXEQ COEFFICIENT SPACE TRIANGULAR MATRIX EXAMPLE 361
FIGURE 4-46: MEASURING $V_{TX\text{-}EIEOS\text{-}FS}$ AND $V_{TX\text{-}EIEOS\text{-}RS}$ 362
FIGURE 4-47: COMPLIANCE PATTERN AND RESULTING PACKAGE LOSS TEST WAVEFORM 363
FIGURE 4-48: TRANSMITTER MARGINING VOLTAGE LEVELS AND CODES 364
FIGURE 4-49: PLOT OF TRANSMITTER HPF FILTER FUNCTIONS 366
FIGURE 4-50: ALGORITHM TO REMOVE DE-EMPHASIS INDUCED JITTER 367
FIGURE 4-51: EXAMPLE OF DE-EMPHASIS JITTER REMOVAL 368
FIGURE 4-52: RELATION BETWEEN DATA EDGE PDFS AND RECOVERED DATA CLOCK 370
FIGURE 4-53: DERIVATION OF $T_{TX\text{-}UTJ}$ AND $T_{TX\text{-}UDJDD}$ 370
FIGURE 4-54: PWJ RELATIVE TO CONSECUTIVE EDGES 1 UI APART 371
FIGURE 4-55: DEFINITION OF $T_{TX\text{-}UPW\text{-}DJDD}$ AND $T_{TX\text{-}UPW\text{-}TJ}$ 372
FIGURE 4-56: TX, RX DIFFERENTIAL RETURN LOSS MASK 372
FIGURE 4-57: TX, RX COMMON MODE RETURN LOSS MASK 373
FIGURE 4-58: CALIBRATION CHANNEL VALIDATION 380
FIGURE 4-59: CALIBRATION CHANNEL SHOWING $T_{MIN\text{-}PULSE}$ 380
FIGURE 4-60: CALIBRATION CHANNEL $|S_{11}|$ PLOT WITH TOLERANCE LIMITS 381
FIGURE 4-61: SETUP FOR CALIBRATING RECEIVER TEST CIRCUIT INTO A REFERENCE LOAD 381
FIGURE 4-62: SETUP FOR TESTING RECEIVER 382
FIGURE 4-63: RECEIVER EYE MARGINS 385
FIGURE 4-64: SIGNAL AT RECEIVER REFERENCE LOAD SHOWING MIN/MAX SWING 386
FIGURE 4-65: RX TESTBOARD TOPOLOGY 387

FIGURE 4-66: INSERTION LOSS GUIDELINES FOR CALIBRATION/BREAKOUT CHANNELS 388
FIGURE 4-67: BEHAVIORAL CDR MODEL FOR RX MEASUREMENT 389
FIGURE 4-68: TRANSFER FUNCTION FOR BEHAVIORAL CTLE 390
FIGURE 4-69: LOSS CURVES FOR BEHAVIORAL CTLE 390
FIGURE 4-70: EQUATION AND FLOW DIAGRAM FOR 1-TAP DFE 391
FIGURE 4-71: SETUP FOR CALIBRATING THE STRESSED VOLTAGE EYE 392
FIGURE 4-72: LAYOUT FOR STRESSED VOLTAGE TESTING OF RECEIVER 394
FIGURE 4-73: LAYOUT FOR CALIBRATING THE STRESSED JITTER EYE 395
FIGURE 4-74: SWEPT SJ MASK 396
FIGURE 4-75: LAYOUT FOR JITTER TESTING COMMON REFCLK RX 397
FIGURE 4-76: LAYOUT FOR JITTER TESTING DATA CLOCKED REFCLK RX 397
FIGURE 4-77: SWEPT SJ MASK 399
FIGURE 4-78: EXIT FROM IDLE VOLTAGE AND TIME MARGINS 403
FIGURE 4-79: A 30 KHZ BEACON SIGNALING THROUGH A 75 NF CAPACITOR 408
FIGURE 4-80: BEACON, WHICH INCLUDES A 2-NS PULSE THROUGH A 75 NF CAPACITOR 408
FIGURE 4-81: SIMULATION ENVIRONMENT FOR CHARACTERIZING CHANNEL 410
FIGURE 4-82: EXTRACTING EYE MARGINS FROM CHANNEL SIMULATION RESULTS 414
FIGURE 4-83: MULTI-SEGMENT CHANNEL EXAMPLE 415
FIGURE 4-84: FLOW DIAGRAM FOR CHANNEL TOLERANCING 416
FIGURE 4-85: TX/RX BEHAVIORAL PACKAGE MODELS 417
FIGURE 4-86: BEHAVIORAL TX AND RX S-PARAMETER FILE DETAILS 417
FIGURE 4-87: DERIVATION OF JITTER PARAMETERS IN TABLE 4-26 420
FIGURE 4-88: EH, EW MASK 420
FIGURE 4-89: REFCLK TEST SETUP 423
FIGURE 4-90: LIMITS FOR PHASE JITTER FROM THE REFERENCE CLOCK 424
FIGURE 4-91: COMMON REFCLK RX ARCHITECTURE 425
FIGURE 4-92: REFCLK TRANSPORT DELAY PATHS FOR A COMMON REFCLK RX ARCHITECTURE 426
FIGURE 4-93: DATA CLOCKED RX ARCHITECTURE 428
FIGURE 4-94: SEPARATE REFCLK WITH NO SSC (SRNS) ARCHITECTURE 430
FIGURE 4-95: SEPARATE REFCLK WITH INDEPENDENT SSC ARCHITECTURE 431
FIGURE 4-96: INFORMATIVE CDR JITTER TRANSFER FUNCTION FOR THE SEPARATE REFCLK WITH INDEPENDENT SSC (SRIS) ARCHITECTURE AT 5.0 GT/S 432
FIGURE 4-97: 8.0 GT/S COMMON REFCLK RX ARCHITECTURE WITH ω_N, ζ LIMITS 435
FIGURE 4-98: 8.0 GT/S DATA CLOCKED RX ARCHITECTURE WITH ω_N, ζ LIMITS 437
FIGURE 4-99: INFORMATIVE CDR JITTER TRANSFER FUNCTION FOR THE SEPARATE REFCLK WITH INDEPENDENT SSC (SRIS) ARCHITECTURE AT 8.0 GT/S 439
FIGURE 5-1: LINK POWER MANAGEMENT STATE FLOW DIAGRAM 445
FIGURE 5-2: ENTRY INTO THE L1 LINK STATE 453
FIGURE 5-3: EXIT FROM L1 LINK STATE INITIATED BY UPSTREAM COMPONENT 456
FIGURE 5-4: CONCEPTUAL DIAGRAMS SHOWING TWO EXAMPLE CASES OF WAKE# ROUTING. 459
FIGURE 5-5: A CONCEPTUAL PME CONTROL STATE MACHINE 463
FIGURE 5-6: L1 TRANSITION SEQUENCE ENDING WITH A REJECTION (L0S ENABLED) 476
FIGURE 5-7: L1 SUCCESSFUL TRANSITION SEQUENCE 477
FIGURE 5-8: EXAMPLE OF L1 EXIT LATENCY COMPUTATION 479
FIGURE 5-9: STATE DIAGRAM FOR L1 PM SUBSTATES 485
FIGURE 5-10: DOWNSTREAM PORT WITH A SINGLE PLL 486

FIGURE 5-11: MULTIPLE DOWNSTREAM PORTS WITH A SHARED PLL 487
FIGURE 5-12: EXAMPLE: L1.1 WAVEFORMS ILLUSTRATING UPSTREAM PORT INITIATED EXIT ... 489
FIGURE 5-13: EXAMPLE: L1.1 WAVEFORMS ILLUSTRATING DOWNSTREAM PORT INITIATED EXIT 490
FIGURE 5-14: L1.2 SUBSTATES 491
FIGURE 5-15: EXAMPLE: ILLUSTRATION OF BOUNDARY CONDITION DUE TO DIFFERENT SAMPLING OF CLKREQ# 492
FIGURE 5-16: EXAMPLE: L1.2 WAVEFORMS ILLUSTRATING UPSTREAM PORT INITIATED EXIT ... 494
FIGURE 5-17: EXAMPLE: L1.2 WAVEFORMS ILLUSTRATING DOWNSTREAM PORT INITIATED EXIT 494
FIGURE 6-1: ERROR CLASSIFICATION 504
FIGURE 6-2: FLOWCHART SHOWING SEQUENCE OF DEVICE ERROR SIGNALING AND LOGGING OPERATIONS 521
FIGURE 6-3: PSEUDO LOGIC DIAGRAM FOR ERROR MESSAGE CONTROLS 522
FIGURE 6-4: TC FILTERING EXAMPLE 542
FIGURE 6-5: TC TO VC MAPPING EXAMPLE 543
FIGURE 6-6: AN EXAMPLE OF TRAFFIC FLOW ILLUSTRATING INGRESS AND EGRESS 544
FIGURE 6-7: AN EXAMPLE OF DIFFERENTIATED TRAFFIC FLOW THROUGH A SWITCH 545
FIGURE 6-8: SWITCH ARBITRATION STRUCTURE 546
FIGURE 6-9: VC ID AND PRIORITY ORDER – AN EXAMPLE 547
FIGURE 6-10: MULTI-FUNCTION ARBITRATION MODEL 550
FIGURE 6-11: ROOT COMPLEX REPRESENTED AS A SINGLE COMPONENT 584
FIGURE 6-12: ROOT COMPLEX REPRESENTED AS MULTIPLE COMPONENTS 585
FIGURE 6-13: EXAMPLE SYSTEM TOPOLOGY WITH ARI DEVICES 599
FIGURE 6-14: SEGMENTATION OF THE MULTICAST ADDRESS RANGE 600
FIGURE 6-15: LATENCY FIELDS FORMAT FOR LTR MESSAGES 618
FIGURE 6-16: CLKREQ# AND CLOCK POWER MANAGEMENT 622
FIGURE 6-17: USE OF LTR AND CLOCK POWER MANAGEMENT 623
FIGURE 6-18: CODES AND EQUIVALENT WAKE# PATTERNS 625
FIGURE 6-19: EXAMPLE PLATFORM TOPOLOGY SHOWING A LINK WHERE OBFF IS CARRIED BY MESSAGES 626
FIGURE 6-20. PASID TLP PREFIX: 629
FIGURE 6-21: SAMPLE SYSTEM BLOCK DIAGRAM 633
FIGURE 6-22: LN PROTOCOL BASIC OPERATION 634
FIGURE 6-23: EXAMPLE SYSTEM TOPOLOGIES USING PTM 640
FIGURE 6-24: PRECISION TIME MEASUREMENT LINK PROTOCOL 641
FIGURE 6-25: PRECISION TIME MEASUREMENT EXAMPLE 643
FIGURE 6-26: PTM REQUESTER OPERATION 646
FIGURE 6-27: PTM TIMESTAMP CAPTURE EXAMPLE 649
FIGURE 7-1: PCI EXPRESS ROOT COMPLEX DEVICE MAPPING 654
FIGURE 7-2: PCI EXPRESS SWITCH DEVICE MAPPING 654
FIGURE 7-3: PCI EXPRESS CONFIGURATION SPACE LAYOUT 655
FIGURE 7-4: COMMON CONFIGURATION SPACE HEADER 666
FIGURE 7-5: TYPE 0 CONFIGURATION SPACE HEADER 673
FIGURE 7-6: TYPE 1 CONFIGURATION SPACE HEADER 675
FIGURE 7-7: POWER MANAGEMENT CAPABILITIES REGISTER 679

FIGURE 7-8: POWER MANAGEMENT STATUS/CONTROL REGISTER 680
FIGURE 7-9: VECTOR CONTROL FOR MSI-X TABLE ENTRIES 681
FIGURE 7-10: PCI EXPRESS CAPABILITY STRUCTURE 683
FIGURE 7-11: PCI EXPRESS CAPABILITY LIST REGISTER 683
FIGURE 7-12: PCI EXPRESS CAPABILITIES REGISTER 684
FIGURE 7-13: DEVICE CAPABILITIES REGISTER 686
FIGURE 7-14: DEVICE CONTROL REGISTER 691
FIGURE 7-15: DEVICE STATUS REGISTER 698
FIGURE 7-16: LINK CAPABILITIES REGISTER 700
FIGURE 7-17: LINK CONTROL REGISTER 705
FIGURE 7-18: LINK STATUS REGISTER 714
FIGURE 7-19: SLOT CAPABILITIES REGISTER 717
FIGURE 7-20: SLOT CONTROL REGISTER 719
FIGURE 7-21: SLOT STATUS REGISTER 723
FIGURE 7-22: ROOT CONTROL REGISTER 725
FIGURE 7-23: ROOT CAPABILITIES REGISTER 726
FIGURE 7-24: ROOT STATUS REGISTER 727
FIGURE 7-25: DEVICE CAPABILITIES 2 REGISTER 728
FIGURE 7-26: DEVICE CONTROL 2 REGISTER 734
FIGURE 7-27: LINK CAPABILITIES 2 REGISTER 738
FIGURE 7-28: LINK CONTROL 2 REGISTER 741
FIGURE 7-29: LINK STATUS 2 REGISTER 746
FIGURE 7-30: PCI EXPRESS EXTENDED CONFIGURATION SPACE LAYOUT 750
FIGURE 7-31: PCI EXPRESS EXTENDED CAPABILITY HEADER 751
FIGURE 7-32: PCI EXPRESS ADVANCED ERROR REPORTING EXTENDED CAPABILITY STRUCTURE 753
FIGURE 7-33: ADVANCED ERROR REPORTING EXTENDED CAPABILITY HEADER 754
FIGURE 7-34: UNCORRECTABLE ERROR STATUS REGISTER 755
FIGURE 7-35: UNCORRECTABLE ERROR MASK REGISTER 757
FIGURE 7-36: UNCORRECTABLE ERROR SEVERITY REGISTER 759
FIGURE 7-37: CORRECTABLE ERROR STATUS REGISTER 761
FIGURE 7-38: CORRECTABLE ERROR MASK REGISTER 762
FIGURE 7-39: ADVANCED ERROR CAPABILITIES AND CONTROL REGISTER 763
FIGURE 7-40: HEADER LOG REGISTER 765
FIGURE 7-41: ROOT ERROR COMMAND REGISTER 765
FIGURE 7-42: ROOT ERROR STATUS REGISTER 767
FIGURE 7-43: ERROR SOURCE IDENTIFICATION REGISTER 769
FIGURE 7-44: TLP PREFIX LOG REGISTER 770
FIGURE 7-45: PCI EXPRESS VIRTUAL CHANNEL CAPABILITY STRUCTURE 771
FIGURE 7-46: VIRTUAL CHANNEL EXTENDED CAPABILITY HEADER 772
FIGURE 7-47: PORT VC CAPABILITY REGISTER 1 773
FIGURE 7-48: PORT VC CAPABILITY REGISTER 2 774
FIGURE 7-49: PORT VC CONTROL REGISTER 775
FIGURE 7-50: PORT VC STATUS REGISTER 776
FIGURE 7-51: VC RESOURCE CAPABILITY REGISTER 777
FIGURE 7-52: VC RESOURCE CONTROL REGISTER 779

FIGURE 7-53: VC RESOURCE STATUS REGISTER 781
FIGURE 7-54: EXAMPLE VC ARBITRATION TABLE WITH 32 PHASES 783
FIGURE 7-55: EXAMPLE PORT ARBITRATION TABLE WITH 128 PHASES AND 2-BIT TABLE ENTRIES 784
FIGURE 7-56: PCI EXPRESS DEVICE SERIAL NUMBER CAPABILITY STRUCTURE 785
FIGURE 7-57: DEVICE SERIAL NUMBER EXTENDED CAPABILITY HEADER 786
FIGURE 7-58: SERIAL NUMBER REGISTER 787
FIGURE 7-59: PCI EXPRESS ROOT COMPLEX LINK DECLARATION CAPABILITY 788
FIGURE 7-60: ROOT COMPLEX LINK DECLARATION EXTENDED CAPABILITY HEADER 789
FIGURE 7-61: ELEMENT SELF DESCRIPTION REGISTER 790
FIGURE 7-62: LINK ENTRY 791
FIGURE 7-63: LINK DESCRIPTION REGISTER 791
FIGURE 7-64: LINK ADDRESS FOR LINK TYPE 0 793
FIGURE 7-65: LINK ADDRESS FOR LINK TYPE 1 794
FIGURE 7-66: ROOT COMPLEX INTERNAL LINK CONTROL CAPABILITY 795
FIGURE 7-67: ROOT INTERNAL LINK CONTROL EXTENDED CAPABILITY HEADER 795
FIGURE 7-68: ROOT COMPLEX LINK CAPABILITIES REGISTER 796
FIGURE 7-69: ROOT COMPLEX LINK CONTROL REGISTER 800
FIGURE 7-70: ROOT COMPLEX LINK STATUS REGISTER 801
FIGURE 7-71: PCI EXPRESS POWER BUDGETING CAPABILITY STRUCTURE 802
FIGURE 7-72: POWER BUDGETING EXTENDED CAPABILITY HEADER 803
FIGURE 7-73: POWER BUDGETING DATA REGISTER 804
FIGURE 7-74: POWER BUDGET CAPABILITY REGISTER 806
FIGURE 7-75: ACS EXTENDED CAPABILITY 807
FIGURE 7-76: ACS EXTENDED CAPABILITY HEADER 807
FIGURE 7-77: ACS CAPABILITY REGISTER 808
FIGURE 7-78: ACS CONTROL REGISTER 809
FIGURE 7-79: EGRESS CONTROL VECTOR REGISTER 811
FIGURE 7-80: ROOT COMPLEX EVENT COLLECTOR ENDPOINT ASSOCIATION CAPABILITY 812
FIGURE 7-81: ROOT COMPLEX EVENT COLLECTOR ENDPOINT ASSOCIATION EXTENDED CAPABILITY HEADER 813
FIGURE 7-82: PCI EXPRESS MFVC CAPABILITY STRUCTURE 814
FIGURE 7-83: MFVC EXTENDED CAPABILITY HEADER 815
FIGURE 7-84: PORT VC CAPABILITY REGISTER 1 816
FIGURE 7-85: PORT VC CAPABILITY REGISTER 2 817
FIGURE 7-86: PORT VC CONTROL REGISTER 818
FIGURE 7-87: PORT VC STATUS REGISTER 819
FIGURE 7-88: VC RESOURCE CAPABILITY REGISTER 819
FIGURE 7-89: VC RESOURCE CONTROL REGISTER 821
FIGURE 7-90: VC RESOURCE STATUS REGISTER 823
FIGURE 7-91: PCI EXPRESS VSEC STRUCTURE 827
FIGURE 7-92: VENDOR-SPECIFIC EXTENDED CAPABILITY HEADER 827
FIGURE 7-93: VENDOR-SPECIFIC HEADER 828
FIGURE 7-94: ROOT COMPLEX FEATURES CAPABILITY STRUCTURE 829
FIGURE 7-95: RCRB HEADER EXTENDED CAPABILITY HEADER 829
FIGURE 7-96: VENDOR ID AND DEVICE ID 830

FIGURE 7-97: RCRB CAPABILITIES 831
FIGURE 7-98: RCRB CONTROL 831
FIGURE 7-99: MULTICAST EXTENDED CAPABILITY STRUCTURE 832
FIGURE 7-100: MULTICAST EXTENDED CAPABILITY HEADER 832
FIGURE 7-101: MULTICAST CAPABILITY REGISTER 833
FIGURE 7-102: MULTICAST CONTROL REGISTER 834
FIGURE 7-103: MC_BASE_ADDRESS REGISTER 835
FIGURE 7-104: MC_RECEIVE REGISTER 835
FIGURE 7-105: MC_BLOCK_ALL REGISTER 836
FIGURE 7-106: MC_BLOCK_UNTRANSLATED REGISTER 837
FIGURE 7-107: MC_OVERLAY_BAR 838
FIGURE 7-108: RESIZABLE BAR CAPABILITY 840
FIGURE 7-109: RESIZABLE BAR EXTENDED CAPABILITY HEADER 840
FIGURE 7-110: RESIZABLE BAR CAPABILITY REGISTER 840
FIGURE 7-111: RESIZABLE BAR CONTROL REGISTER 842
FIGURE 7-112: ARI CAPABILITY 843
FIGURE 7-113: ARI CAPABILITY HEADER 844
FIGURE 7-114: ARI CAPABILITY REGISTER 844
FIGURE 7-115: ARI CONTROL REGISTER 845
FIGURE 7-116: DYNAMIC POWER ALLOCATION CAPABILITY STRUCTURE 846
FIGURE 7-117: DPA EXTENDED CAPABILITY HEADER 846
FIGURE 7-118: DPA CAPABILITY REGISTER 847
FIGURE 7-119: DPA LATENCY INDICATOR REGISTER 848
FIGURE 7-120: DPA STATUS REGISTER 848
FIGURE 7-121: DPA CONTROL REGISTER 849
FIGURE 7-122: DPA POWER ALLOCATION ARRAY 849
FIGURE 7-123: LTR EXTENDED CAPABILITY STRUCTURE 850
FIGURE 7-124: LTR EXTENDED CAPABILITY HEADER 850
FIGURE 7-125: MAX SNOOP LATENCY REGISTER 851
FIGURE 7-126: MAX NO-SNOOP LATENCY REGISTER 851
FIGURE 7-127: TPH EXTENDED CAPABILITY STRUCTURE 852
FIGURE 7-128: TPH REQUESTER EXTENDED CAPABILITY HEADER 853
FIGURE 7-129: TPH REQUESTER CAPABILITY REGISTER 853
FIGURE 7-130: TPH REQUESTER CONTROL REGISTER 855
FIGURE 7-131: TPH ST TABLE 856
FIGURE 7-132: SECONDARY PCI EXPRESS EXTENDED CAPABILITY STRUCTURE 857
FIGURE 7-133: SECONDARY PCI EXPRESS EXTENDED CAPABILITY HEADER 857
FIGURE 7-134: LINK CONTROL 3 REGISTER 858
FIGURE 7-135: LANE ERROR STATUS REGISTER 859
FIGURE 7-136: LANE EQUALIZATION CONTROL REGISTER 860
FIGURE 7-137: LANE ((MAXIMUM LINK WIDTH – 1):0) EQUALIZATION CONTROL REGISTER 861
FIGURE 7-138. M-PCIE EXTENDED CAPABILITY 864
FIGURE 7-139. M-PCIE EXTENDED CAPABILITY HEADER 864
FIGURE 7-140: M-PCIE CAPABILITIES REGISTER 866
FIGURE 7-141: M-PCIE CONTROL REGISTER 867
FIGURE 7-142: M-PCIE STATUS REGISTER 868

FIGURE 7-143: M-PCIE LANE ERROR STATUS REGISTER 869
FIGURE 7-144: M-PCIE PHY CONTROL ADDRESS REGISTER 870
FIGURE 7-145: M-PCIE PHY CONTROL DATA REGISTER 871
FIGURE 7-146: LNR EXTENDED CAPABILITY 876
FIGURE 7-147: LNR EXTENDED CAPABILITY HEADER 876
FIGURE 7-148: LNR CAPABILITY REGISTER 877
FIGURE 7-149: LNR CONTROL REGISTER 878
FIGURE 7-150: DPC EXTENDED CAPABILITY 879
FIGURE 7-151: DPC EXTENDED CAPABILITY HEADER 880
FIGURE 7-152: DPC CAPABILITY REGISTER 881
FIGURE 7-153: DPC CONTROL REGISTER 883
FIGURE 7-154. DPC STATUS REGISTER 885
FIGURE 7-155. DPC ERROR SOURCE ID REGISTER 887
FIGURE 7-156: RP PIO STATUS REGISTER 888
FIGURE 7-157: RP PIO MASK REGISTER 889
FIGURE 7-158: RP PIO SEVERITY REGISTER 890
FIGURE 7-159: RP PIO SYSERROR REGISTER 891
FIGURE 7-160: RP PIO EXCEPTION REGISTER 892
FIGURE 7-161: RP PIO HEADER LOG REGISTER 893
FIGURE 7-162: RP PIO IMPSPEC LOG REGISTER 893
FIGURE 7-163: RP PIO TLP PREFIX LOG REGISTER 894
FIGURE 7-164: PTM EXTENDED CAPABILITY STRUCTURE 896
FIGURE 7-165: PTM EXTENDED CAPABILITY HEADER 896
FIGURE 7-166: PTM CAPABILITY REGISTER 897
FIGURE 7-167: PTM CONTROL REGISTER 898
FIGURE 7-168: L1 PM SUBSTATES CAPABILITY 900
FIGURE 7-169: L1 PM SUBSTATES EXTENDED CAPABILITY HEADER 900
FIGURE 7-170: L1 PM SUBSTATES CAPABILITIES REGISTER 901
FIGURE 7-171: L1 PM SUBSTATES CONTROL 1 REGISTER 903
FIGURE 7-172: L1 PM SUBSTATES CONTROL 2 REGISTER 905
FIGURE 7-173: FRS EXTENDED CAPABILITY 906
FIGURE 7-174: FRS QUEUING EXTENDED CAPABILITY HEADER 907
FIGURE 7-175: FRS QUEUING CAPABILITY REGISTER 908
FIGURE 7-176: FRS QUEUEING STATUS REGISTER 909
FIGURE 7-177: FRS QUEUEING CONTROL REGISTER 909
FIGURE 7-178: FRS MESSAGE QUEUE REGISTER 910
FIGURE 7-179: READINESS TIME REPORTING EXTENDED CAPABILITY 912
FIGURE 7-180: READINESS TIME ENCODING 912
FIGURE 7-181: READINESS TIME REPORTING EXTENDED CAPABILITY HEADER 913
FIGURE 7-182: READINESS TIME REPORTING 1 REGISTER 914
FIGURE 7-183: READINESS TIME REPORTING 2 REGISTER 915
FIGURE 8-1: ADAPTATION OF MIPI-M-PHY 916
FIGURE 8-2: M-PCIE LINK EXAMPLE. 917
FIGURE 8-3: RRAP WRITE COMMAND FORMAT 933
FIGURE 8-4: RRAP WRITE RESPONSE FORMAT 934
FIGURE 8-5: RRAP READ COMMAND FORMAT 934

FIGURE 8-6: RRAP READ RESPONSE FORMAT 934
FIGURE 8-7: M-PHY TEST BLOCK 942
FIGURE 8-8: EXAMPLE TRANSMIT COMPLIANCE MODE 943
FIGURE 8-9: EXAMPLE OF A CONFIGURATION TO ENABLE ANALOG LOOPBACK MODE 944
FIGURE 8-10 : EXAMPLE ANALOG LOOPBACK MODE CONFIGURATION 945
FIGURE 8-11: M-PCIE LTSSM 957
FIGURE 8-12: CONFIGURATION SUB-STATE MACHINE 959
FIGURE 8-13: RECOVERY SUB-STATE MACHINE 966
FIGURE 8-14: EXAMPLE OF DOWNSTREAM PORT INITIATED LINK RE-CONFIGURATION 980
FIGURE A-1: AN EXAMPLE SHOWING ENDPOINT-TO-ROOT-COMPLEX AND PEER-TO-PEER COMMUNICATION MODELS 994
FIGURE A-2: TWO BASIC BANDWIDTH RESOURCING PROBLEMS: OVER-SUBSCRIPTION AND CONGESTION 995
FIGURE A-3: A SIMPLIFIED EXAMPLE ILLUSTRATING PCI EXPRESS ISOCHRONOUS PARAMETERS 1000
FIGURE C-1: SCRAMBLING SPECTRUM AT 2.5 GT/S FOR DATA VALUE OF 0 1019
FIGURE E-1: REFERENCE TOPOLOGY FOR IDO USE 1026
FIGURE G-1: DEVICE AND PROCESSOR CONNECTED USING A PMUX LINK 1034
FIGURE G-2: PMUX LINK 1035
FIGURE G-3: PMUX PACKET FLOW THROUGH THE LAYERS 1036
FIGURE G-4: PMUX PACKET 1043
FIGURE G-5: TLP AND PMUX PACKET FRAMING (8B10B ENCODING) 1044
FIGURE G-6: TLP AND PMUX PACKET FRAMING (128B/130B ENCODING) 1046
FIGURE G-7: PMUX EXTENDED CAPABILITY 1050
FIGURE G-8: PMUX EXTENDED CAPABILITY HEADER 1050
FIGURE G-9: PMUX CAPABILITY REGISTER 1051
FIGURE G-10: PMUX CONTROL REGISTER 1053
FIGURE G-11: PMUX STATUS REGISTER 1054
FIGURE G-12: PMUX PROTOCOL ARRAY ENTRY 1057
FIGURE H-1: RESET COMPLETION TO L0 TIMING DIAGRAM 1060
FIGURE H-2: L0 TO TRANSMITTER IN STALL TIMING DIAGRAM 1061
FIGURE H-3: DSP INITIATED L0 TO L1 ENTRY TIMING DIAGRAM 1062
FIGURE H-4: DOWNSTREAM PORT INITIATED LINK BANDWIDTH CHANGE TIMING DIAGRAM . 1063
FIGURE H-5: UPSTREAM PORT INITIATED LINK BANDWIDTH CHANGE TIMING DIAGRAM 1064
FIGURE I-1: CONTINUOUS MODE CRPAT (LANE 0 SHOWN) 1066
FIGURE I-2: BURST MODE CRPAT (LANE 0 SHOWN) 1067

Tables

TABLE 2-1: TRANSACTION TYPES FOR DIFFERENT ADDRESS SPACES 63
TABLE 2-2: FMT[2:0] FIELD VALUES 68
TABLE 2-3: FMT[2:0] AND TYPE[4:0] FIELD ENCODINGS 69
TABLE 2-4: LENGTH[9:0] FIELD ENCODING 70
TABLE 2-5: ADDRESS TYPE (AT) FIELD ENCODINGS 75
TABLE 2-6: ADDRESS FIELD MAPPING 76
TABLE 2-7: HEADER FIELD LOCATIONS FOR NON-ARI ID ROUTING 77
TABLE 2-8: HEADER FIELD LOCATIONS FOR ARI ID ROUTING 77
TABLE 2-9: BYTE ENABLES LOCATION AND CORRESPONDENCE 80
TABLE 2-10: ORDERING ATTRIBUTES 85
TABLE 2-11: CACHE COHERENCY MANAGEMENT ATTRIBUTE 86
TABLE 2-12: DEFINITION OF TC FIELD ENCODINGS 86
TABLE 2-13: LENGTH FIELD VALUES FOR ATOMICOP REQUESTS 87
TABLE 2-14: TPH TLP PREFIX BIT MAPPING 91
TABLE 2-15: LOCATION OF PH[1:0] IN TLP HEADER 92
TABLE 2-16: PROCESSING HINT ENCODING 92
TABLE 2-17: LOCATION OF ST[7:0] IN TLP HEADERS 93
TABLE 2-18: MESSAGE ROUTING 95
TABLE 2-19: INTX MECHANISM MESSAGES 97
TABLE 2-20: BRIDGE MAPPING FOR INTX VIRTUAL WIRES 99
TABLE 2-21: POWER MANAGEMENT MESSAGES 101
TABLE 2-22: ERROR SIGNALING MESSAGES 102
TABLE 2-23: UNLOCK MESSAGE 103
TABLE 2-24: SET_SLOT_POWER_LIMIT MESSAGE 103
TABLE 2-25: VENDOR_DEFINED MESSAGES 104
TABLE 2-26: NOTIFICATION REASON (NR) FIELD ENCODINGS 107
TABLE 2-27: LN MESSAGES 107
TABLE 2-28: DRS MESSAGE 109
TABLE 2-29: FRS MESSAGE 110
TABLE 2-30: IGNORED MESSAGES 111
TABLE 2-31: LTR MESSAGE 112
TABLE 2-32: OBFF MESSAGE 113
TABLE 2-33: PRECISION TIME MEASUREMENT MESSAGES 114
TABLE 2-34: COMPLETION STATUS FIELD VALUES 116
TABLE 2-35: LOCAL TLP PREFIX TYPES 119
TABLE 2-36: END-END TLP PREFIX TYPES 120
TABLE 2-37: CALCULATING BYTE COUNT FROM LENGTH AND BYTE ENABLES 136
TABLE 2-38: CALCULATING LOWER ADDRESS FROM 1ST DW BE 137
TABLE 2-39: ORDERING RULES SUMMARY 142
TABLE 2-40: TC TO VC MAPPING EXAMPLE 150
TABLE 2-41: FLOW CONTROL CREDIT TYPES 154
TABLE 2-42: TLP FLOW CONTROL CREDIT CONSUMPTION 154
TABLE 2-43: MINIMUM INITIAL FLOW CONTROL ADVERTISEMENTS 155

TABLE 2-44: UPDATEFC TRANSMISSION LATENCY GUIDELINES FOR 2.5 GT/S MODE OPERATION BY LINK WIDTH AND MAX PAYLOAD (SYMBOL TIMES) 163
TABLE 2-45: UPDATEFC TRANSMISSION LATENCY GUIDELINES FOR 5.0 GT/S MODE OPERATION BY LINK WIDTH AND MAX PAYLOAD (SYMBOL TIMES) 163
TABLE 2-46: UPDATEFC TRANSMISSION LATENCY GUIDELINES FOR 8.0 GT/S OPERATION BY LINK WIDTH AND MAX PAYLOAD (SYMBOL TIMES) 163
TABLE 2-47: MAPPING OF BITS INTO ECRC FIELD 165
TABLE 3-1: DLLP TYPE ENCODINGS 185
TABLE 3-2: MAPPING OF BITS INTO CRC FIELD 188
TABLE 3-3: MAPPING OF BITS INTO LCRC FIELD 192
TABLE 3-4: UNADJUSTED REPLAY_TIMER LIMITS FOR 2.5 GT/S MODE OPERATION BY LINK WIDTH AND MAX_PAYLOAD_SIZE (SYMBOL TIMES) TOLERANCE: -0%/+100% 197
TABLE 3-5: UNADJUSTED REPLAY_TIMER LIMITS FOR 5.0 GT/S MODE OPERATION BY LINK WIDTH AND MAX_PAYLOAD_SIZE (SYMBOL TIMES) TOLERANCE: -0%/+100% 197
TABLE 3-6: UNADJUSTED REPLAY_TIMER LIMITS FOR 8.0 GT/S OPERATION BY LINK WIDTH AND MAX_PAYLOAD_SIZE (SYMBOL TIMES) TOLERANCE: -0%/+100% 198
TABLE 3-7: ACK TRANSMISSION LATENCY LIMIT AND ACKFACTOR FOR 2.5 GT/S MODE OPERATION BY LINK WIDTH AND MAX PAYLOAD (SYMBOL TIMES) 208
TABLE 3-8: ACK TRANSMISSION LATENCY LIMIT AND ACKFACTOR FOR 5.0 GT/S MODE OPERATION BY LINK WIDTH AND MAX PAYLOAD (SYMBOL TIMES) 209
TABLE 3-9: ACK TRANSMISSION LATENCY LIMIT AND ACKFACTOR FOR 8.0 GT/S MODE OPERATION BY LINK WIDTH AND MAX PAYLOAD (SYMBOL TIMES) 209
TABLE 4-1: SPECIAL SYMBOLS 214
TABLE 4-2: FRAMING TOKEN ENCODING 224
TABLE 4-3: TRANSMITTER PRESET ENCODING 245
TABLE 4-4: RECEIVER PRESET HINT ENCODING 245
TABLE 4-5: TS1 ORDERED SET 248
TABLE 4-6: TS2 ORDERED SET 251
TABLE 4-7: ELECTRICAL IDLE ORDERED SET (EIOS) FOR 2.5 GT/S AND 5.0 GT/S DATA RATES 253
TABLE 4-8: ELECTRICAL IDLE ORDERED SET (EIOS) FOR 8.0 GT/S AND ABOVE DATA RATES .. 253
TABLE 4-9: ELECTRICAL IDLE EXIT ORDERED SET (EIEOS) FOR 5.0 GT/S DATA RATE 253
TABLE 4-10: ELECTRICAL IDLE EXIT ORDERED SET (EIEOS) FOR 8.0 GT/S AND ABOVE DATA RATES 253
TABLE 4-11: ELECTRICAL IDLE INFERENCE CONDITIONS 256
TABLE 4-12: FTS FOR 8.0 GT/S AND ABOVE DATA RATES 259
TABLE 4-13: SDS ORDERED SET (FOR 8.0 GT/S AND ABOVE DATA RATE) 260
TABLE 4-14: LINK STATUS MAPPED TO THE LTSSM 268
TABLE 4-15: SKP ORDERED SET WITH 128B/130B ENCODING 334
TABLE 4-16: TX PRESET RATIOS AND CORRESPONDING COEFFICIENT VALUES 358
TABLE 4-17: PRESET MEASUREMENT CROSS REFERENCE TABLE 359
TABLE 4-18: TRANSMITTER SPECIFICATIONS 374
TABLE 4-19: 8.0 GT/S SPECIFIC TX VOLTAGE AND JITTER PARAMETERS 377
TABLE 4-20: 5.0 GT/S TOLERANCING LIMITS FOR COMMON REFCLK RX ARCHITECTURE 383
TABLE 4-21: 5.0 GT/S TOLERANCING LIMITS FOR DATA CLOCKED RX ARCHITECTURE 384
TABLE 4-22: STRESSED VOLTAGE EYE PARAMETERS 393
TABLE 4-23: STRESSED JITTER EYE PARAMETERS 395

TABLE 4-24: RECEIVER SPECIFICATIONS 400
TABLE 4-25: WORST CASE TX CORNERS FOR 5.0 GT/S CHANNEL SIMULATION 411
TABLE 4-26: JITTER/VOLTAGE PARAMETERS FOR CHANNEL TOLERANCING 419
TABLE 4-27: CHANNEL TOLERANCING EYE MASK VALUES 421
TABLE 4-28: EIEOS SIGNALING PARAMETERS 422
TABLE 4-29: LIMITS FOR PHASE JITTER FROM THE REFERENCE CLOCK 423
TABLE 4-30: FILTERING FUNCTIONS APPLIED TO REFCLK MEASUREMENTS 425
TABLE 4-31: DIFFERENCE FUNCTION PARAMETERS APPLIED TO REFCLK MEASUREMENT 427
TABLE 4-32: REFCLK PARAMETERS FOR COMMON REFCLK RX ARCHITECTURE AT 5.0 GT/S 427
TABLE 4-33: PLL PARAMETERS FOR DATA CLOCKED RX ARCHITECTURE 429
TABLE 4-34: REFCLK PARAMETERS FOR DATA CLOCKED RX ARCHITECTURE 429
TABLE 4-35: PARAMETERS FOR SEPARATE REFCLK WITH INDEPENDENT SSC ARCHITECTURE (SRIS) AT 5.0 GT/S 433
TABLE 4-36: INFORMATIVE TOTAL JITTER BUDGET FOR 5.0 GT/S SIGNALING FOR SEPARATE REFCLK WITH INDEPENDENT SSC (SRIS) ARCHITECTURE 433
TABLE 4-37: PARAMETERS FOR COMMON REFCLK RX ARCHITECTURE AT 8.0 GT/S 436
TABLE 4-38: PARAMETERS FOR DATA CLOCKED RX ARCHITECTURE AT 8.0 GT/S 438
TABLE 4-39: PARAMETERS FOR SEPARATE REFCLK WITH INDEPENDENT SSC (SRIS) ARCHITECTURE AT 8.0 GT/S 440
TABLE 5-1: SUMMARY OF PCI EXPRESS LINK POWER MANAGEMENT STATES 447
TABLE 5-2: RELATION BETWEEN POWER MANAGEMENT STATES OF LINK AND COMPONENTS 452
TABLE 5-3: ENCODING OF THE ASPM SUPPORT FIELD 480
TABLE 5-4: DESCRIPTION OF THE SLOT CLOCK CONFIGURATION BIT 481
TABLE 5-5: DESCRIPTION OF THE COMMON CLOCK CONFIGURATION BIT 481
TABLE 5-6: ENCODING OF THE L0S EXIT LATENCY FIELD 481
TABLE 5-7: ENCODING OF THE L1 EXIT LATENCY FIELD 482
TABLE 5-8: ENCODING OF THE ENDPOINT L0S ACCEPTABLE LATENCY FIELD 482
TABLE 5-9: ENCODING OF THE ENDPOINT L1 ACCEPTABLE LATENCY FIELD 482
TABLE 5-10: ENCODING OF THE ASPM CONTROL FIELD 483
TABLE 5-11: TIMING PARAMETERS 496
TABLE 5-12: POWER MANAGEMENT SYSTEM MESSAGES AND DLLPS 498
TABLE 6-1: ERROR MESSAGES 507
TABLE 6-2: GENERAL PCI EXPRESS ERROR LIST 523
TABLE 6-3: PHYSICAL LAYER ERROR LIST 524
TABLE 6-4: DATA LINK LAYER ERROR LIST 524
TABLE 6-5: TRANSACTION LAYER ERROR LIST 525
TABLE 6-6: ELEMENTS OF HOT-PLUG 566
TABLE 6-7: ATTENTION INDICATOR STATES 567
TABLE 6-8: POWER INDICATOR STATES 568
TABLE 6-9: ACS P2P REQUEST REDIRECT AND ACS P2P EGRESS CONTROL INTERACTIONS 592
TABLE 6-10: ECRC RULES FOR MC_OVERLAY 605
TABLE 6-11: PROCESSING HINT MAPPING 615
TABLE 6-12: ST MODES OF OPERATION 616
TABLE 6-13. PASID TLP PREFIX 629
TABLE 7-1: ENHANCED CONFIGURATION ADDRESS MAPPING 657
TABLE 7-2: REGISTER AND REGISTER BIT-FIELD TYPES 664

TABLE 7-3: COMMAND REGISTER 667
TABLE 7-4: STATUS REGISTER 669
TABLE 7-5: HEADER TYPE REGISTER 671
TABLE 7-6: SECONDARY STATUS REGISTER 676
TABLE 7-7: BRIDGE CONTROL REGISTER 678
TABLE 7-8: POWER MANAGEMENT CAPABILITIES REGISTER ADDED REQUIREMENTS 679
TABLE 7-9: POWER MANAGEMENT STATUS/CONTROL REGISTER ADDED REQUIREMENTS 680
TABLE 7-10: VECTOR CONTROL FOR MSI-X TABLE ENTRIES 681
TABLE 7-11: PCI EXPRESS CAPABILITY LIST REGISTER 684
TABLE 7-12: PCI EXPRESS CAPABILITIES REGISTER 684
TABLE 7-13: DEVICE CAPABILITIES REGISTER 687
TABLE 7-14: DEVICE CONTROL REGISTER 692
TABLE 7-15: DEVICE STATUS REGISTER 698
TABLE 7-16: LINK CAPABILITIES REGISTER 700
TABLE 7-17: LINK CONTROL REGISTER 706
TABLE 7-18: LINK STATUS REGISTER 714
TABLE 7-19: SLOT CAPABILITIES REGISTER 717
TABLE 7-20: SLOT CONTROL REGISTER 720
TABLE 7-21: SLOT STATUS REGISTER 723
TABLE 7-22: ROOT CONTROL REGISTER 725
TABLE 7-23: ROOT CAPABILITIES REGISTER 726
TABLE 7-24: ROOT STATUS REGISTER 727
TABLE 7-25: DEVICE CAPABILITIES 2 REGISTER 729
TABLE 7-26: DEVICE CONTROL 2 REGISTER 735
TABLE 7-27: LINK CAPABILITIES 2 REGISTER 739
TABLE 7-28: LINK CONTROL 2 REGISTER 742
TABLE 7-29: LINK STATUS 2 REGISTER 747
TABLE 7-30: PCI EXPRESS EXTENDED CAPABILITY HEADER 751
TABLE 7-31: ADVANCED ERROR REPORTING EXTENDED CAPABILITY HEADER 754
TABLE 7-32: UNCORRECTABLE ERROR STATUS REGISTER 755
TABLE 7-33: UNCORRECTABLE ERROR MASK REGISTER 758
TABLE 7-34: UNCORRECTABLE ERROR SEVERITY REGISTER 759
TABLE 7-35: CORRECTABLE ERROR STATUS REGISTER 761
TABLE 7-36: CORRECTABLE ERROR MASK REGISTER 762
TABLE 7-37: ADVANCED ERROR CAPABILITIES AND CONTROL REGISTER 763
TABLE 7-38: HEADER LOG REGISTER 765
TABLE 7-39: ROOT ERROR COMMAND REGISTER 766
TABLE 7-40: ROOT ERROR STATUS REGISTER 767
TABLE 7-41: ERROR SOURCE IDENTIFICATION REGISTER 769
TABLE 7-42: TLP PREFIX LOG REGISTER 770
TABLE 7-43: VIRTUAL CHANNEL EXTENDED CAPABILITY HEADER 772
TABLE 7-44: PORT VC CAPABILITY REGISTER 1 773
TABLE 7-45: PORT VC CAPABILITY REGISTER 2 775
TABLE 7-46: PORT VC CONTROL REGISTER 776
TABLE 7-47: PORT VC STATUS REGISTER 777
TABLE 7-48: VC RESOURCE CAPABILITY REGISTER 778

TABLE 7-49: VC RESOURCE CONTROL REGISTER 779
TABLE 7-50: VC RESOURCE STATUS REGISTER 782
TABLE 7-51: DEFINITION OF THE 4-BIT ENTRIES IN THE VC ARBITRATION TABLE 783
TABLE 7-52: LENGTH OF THE VC ARBITRATION TABLE 783
TABLE 7-53: LENGTH OF PORT ARBITRATION TABLE 785
TABLE 7-54: DEVICE SERIAL NUMBER EXTENDED CAPABILITY HEADER 786
TABLE 7-55: SERIAL NUMBER REGISTER 787
TABLE 7-56: ROOT COMPLEX LINK DECLARATION EXTENDED CAPABILITY HEADER 789
TABLE 7-57: ELEMENT SELF DESCRIPTION REGISTER 790
TABLE 7-58: LINK DESCRIPTION REGISTER 792
TABLE 7-59: LINK ADDRESS FOR LINK TYPE 1 794
TABLE 7-60: ROOT COMPLEX INTERNAL LINK CONTROL EXTENDED CAPABILITY HEADER 796
TABLE 7-61: ROOT COMPLEX LINK CAPABILITIES REGISTER 797
TABLE 7-62: ROOT COMPLEX LINK CONTROL REGISTER 800
TABLE 7-63: ROOT COMPLEX LINK STATUS REGISTER 801
TABLE 7-64: POWER BUDGETING EXTENDED CAPABILITY HEADER 803
TABLE 7-65: POWER BUDGETING DATA REGISTER 804
TABLE 7-66: POWER BUDGET CAPABILITY REGISTER 806
TABLE 7-67: ACS EXTENDED CAPABILITY HEADER 807
TABLE 7-68: ACS CAPABILITY REGISTER 808
TABLE 7-69: ACS CONTROL REGISTER 809
TABLE 7-70: EGRESS CONTROL VECTOR 811
TABLE 7-71: ROOT COMPLEX EVENT COLLECTOR ENDPOINT ASSOCIATION EXTENDED CAPABILITY HEADER 813
TABLE 7-72: MFVC EXTENDED CAPABILITY HEADER 815
TABLE 7-73: PORT VC CAPABILITY REGISTER 1 816
TABLE 7-74: PORT VC CAPABILITY REGISTER 2 817
TABLE 7-75: PORT VC CONTROL REGISTER 818
TABLE 7-76: PORT VC STATUS REGISTER 819
TABLE 7-77: VC RESOURCE CAPABILITY REGISTER 820
TABLE 7-78: VC RESOURCE CONTROL REGISTER 821
TABLE 7-79: VC RESOURCE STATUS REGISTER 824
TABLE 7-80: LENGTH OF FUNCTION ARBITRATION TABLE 826
TABLE 7-81: VENDOR-SPECIFIC EXTENDED CAPABILITY HEADER 827
TABLE 7-82: VENDOR-SPECIFIC HEADER 828
TABLE 7-83: RCRB HEADER EXTENDED CAPABILITY HEADER 830
TABLE 7-84: VENDOR ID AND DEVICE ID 830
TABLE 7-85: RCRB CAPABILITIES 831
TABLE 7-86: RCRB CONTROL 831
TABLE 7-87: MULTICAST EXTENDED CAPABILITY HEADER 833
TABLE 7-88: MULTICAST CAPABILITY REGISTER 834
TABLE 7-89: MULTICAST CONTROL REGISTER 834
TABLE 7-90: MC_BASE ADDRESS REGISTER 835
TABLE 7-91: MC_RECEIVE REGISTER 836
TABLE 7-92: MC_BLOCK_ALL REGISTER 836
TABLE 7-93: MC_BLOCK_UNTRANSLATED REGISTER 837

TABLE 7-94: MC_OVERLAY BAR 838
TABLE 7-95: RESIZABLE BAR EXTENDED CAPABILITY HEADER 840
TABLE 7-96: RESIZABLE BAR CAPABILITY REGISTER 841
TABLE 7-97: RESIZABLE BAR CONTROL REGISTER 842
TABLE 7-98: ARI CAPABILITY HEADER 844
TABLE 7-99: ARI CAPABILITY REGISTER 844
TABLE 7-100: ARI CONTROL REGISTER 845
TABLE 7-101: DPA EXTENDED CAPABILITY HEADER 846
TABLE 7-102: DPA CAPABILITY REGISTER 847
TABLE 7-103: DPA LATENCY INDICATOR REGISTER 848
TABLE 7-104: DPA STATUS REGISTER 848
TABLE 7-105: DPA CONTROL REGISTER 849
TABLE 7-106: SUBSTATE POWER ALLOCATION REGISTER (0 TO SUBSTATE_MAX) 849
TABLE 7-107: LTR EXTENDED CAPABILITY HEADER 850
TABLE 7-108: MAX SNOOP LATENCY REGISTER 851
TABLE 7-109: MAX NO-SNOOP LATENCY REGISTER 852
TABLE 7-110: TPH REQUESTER EXTENDED CAPABILITY HEADER 853
TABLE 7-111: TPH REQUESTER CAPABILITY REGISTER 854
TABLE 7-112: TPH REQUESTER CONTROL REGISTER 855
TABLE 7-113: TPH ST TABLE 856
TABLE 7-114: SECONDARY PCI EXPRESS EXTENDED CAPABILITY HEADER 858
TABLE 7-115: LINK CONTROL 3 REGISTER 858
TABLE 7-116: LANE ERROR STATUS REGISTER 860
TABLE 7-117: LANE ((MAXIMUM LINK WIDTH – 1):0) EQUALIZATION CONTROL REGISTER 861
TABLE 7-118: M-PCIE EXTENDED CAPABILITY HEADER 865
TABLE 7-119: M-PCIE CAPABILITIES REGISTER 866
TABLE 7-120: M-PCIE CONTROL REGISTER 867
TABLE 7-121: M-PCIE STATUS REGISTER 868
TABLE 7-122: M-PCIE LANE ERROR STATUS REGISTER 869
TABLE 7-123: M-PCIE PHY CONTROL ADDRESS REGISTER 870
TABLE 7-124: M-PCIE PHY CONTROL DATA REGISTER 871
TABLE 7-125: PASID EXTENDED CAPABILITY HEADER 873
TABLE 7-126: PASID CAPABILITY REGISTER 874
TABLE 7-127: PASID CONTROL REGISTER 875
TABLE 7-128: LNR EXTENDED CAPABILITY HEADER 876
TABLE 7-129: LNR CAPABILITY REGISTER 877
TABLE 7-130: LNR CONTROL REGISTER 878
TABLE 7-131: DPC EXTENDED CAPABILITY HEADER 880
TABLE 7-132: DPC CAPABILITY REGISTER 881
TABLE 7-133: DPC CONTROL REGISTER 883
TABLE 7-134. DPC STATUS REGISTER 885
TABLE 7-135. DPC ERROR SOURCE ID REGISTER 887
TABLE 7-136: RP PIO STATUS REGISTER 888
TABLE 7-137: RP PIO MASK REGISTER 889
TABLE 7-138: RP PIO SEVERITY REGISTER 890
TABLE 7-139: RP PIO SYSERROR REGISTER 891

TABLE 7-140: RP PIO EXCEPTION REGISTER 892
TABLE 7-141: RP PIO HEADER LOG REGISTER 893
TABLE 7-142: RP PIO IMPSPEC LOG REGISTER 893
TABLE 7-143: RP PIO TLP PREFIX LOG REGISTER 894
TABLE 7-144: PTM EXTENDED CAPABILITY HEADER 896
TABLE 7-145: PTM CAPABILITY REGISTER 897
TABLE 7-146: PTM CONTROL REGISTER 898
TABLE 7-147: L1 PM SUBSTATES EXTENDED CAPABILITY HEADER 900
TABLE 7-148: L1 PM SUBSTATES CAPABILITIES REGISTER 901
TABLE 7-149: L1 PM SUBSTATES CONTROL 1 REGISTER 903
TABLE 7-150: L1 PM SUBSTATES CONTROL 2 REGISTER 905
TABLE 7-151: FRS QUEUING EXTENDED CAPABILITY HEADER 907
TABLE 7-152: FRS QUEUING CAPABILITY REGISTER 908
TABLE 7-153: FRS QUEUING STATUS REGISTER 909
TABLE 7-154: FRS QUEUING CONTROL REGISTER 910
TABLE 7-155: FRS MESSAGE QUEUE REGISTER 910
TABLE 7-156: READINESS TIME REPORTING EXTENDED CAPABILITY HEADER 913
TABLE 7-157: READINESS TIME REPORTING 1 REGISTER 914
TABLE 7-158: READINESS TIME REPORTING 2 REGISTER 915
TABLE 8-1: TABLE SPEED AND RATE CAPABILITIES 919
TABLE 8-2: M-PCIE CAPABILITY ATTRIBUTES 921
TABLE 8-3: M-PCIE CONFIGURATION ATTRIBUTES 929
TABLE 8-4: M-PCIE STATUS ATTRIBUTES 933
TABLE 8-5: RRAP TIMING PARAMETERS 935
TABLE 8-6: RRAP M-PCIE ADDRESS MAP 936
TABLE 8-7: TEST ADDRESS MAP 938
TABLE 8-8: M-PCIE LINE ENCODING 946
TABLE 8-9: TS1 ORDERED SET. 948
TABLE 8-10: TS2 ORDERED SET. 951
TABLE 8-11: M-PCIE ELECTRICAL IDLE INFERENCE CONDITIONS 954
TABLE 8-12: LANE-TO-LANE DE-SKEW TIMING PARAMETERS 955
TABLE 8-13: HIBERN8 TIMING PARAMETERS 977
TABLE 8-14: RECEIVER ERROR 978
TABLE 8-15: M-TX CAPABILITY ATTRIBUTES 982
TABLE 8-16: M-RX CAPABILITY ATTRIBUTES 985
TABLE 8-17: M-TX CONFIGURATION ATTRIBUTES 990
TABLE A-1: ISOCHRONOUS BANDWIDTH RANGES AND GRANULARITIES 997
TABLE B-1: 8B/10B DATA SYMBOL CODES 1005
TABLE B-2: 8B/10B SPECIAL CHARACTER SYMBOL CODES 1013
TABLE F-1: MESSAGE CODE USAGE 1032
TABLE F-2: PCI-SIG-DEFINED VDM SUBTYPE USAGE 1033
TABLE G-1: PCI EXPRESS ATTRIBUTE IMPACT ON PROTOCOL MULTIPLEXING 1037
TABLE G-2: PMUX ATTRIBUTE IMPACT ON PCI EXPRESS 1041
TABLE G-3: PMUX PACKET LAYOUT (8B10B ENCODING) 1045
TABLE G-4: PMUX PACKET LAYOUT (128B/130B ENCODING) 1047
TABLE G-5: SYMBOL 1 BITS [6:3] 1048

TABLE G-6: PMUX EXTENDED CAPABILITY HEADER 1050
TABLE G-7: PMUX CAPABILITY REGISTER 1051
TABLE G-8: PMUX CONTROL REGISTER 1053
TABLE G-9: PMUX STATUS REGISTER 1055
TABLE G-10: PMUX PROTOCOL ARRAY ENTRY 1057
TABLE I-1: RPAT DEFINITION 1065
TABLE I-2: RPAT SHIFTED BY 2 BYTES PER LANE 1066

This page left intentionally blank.

Objective of the Specification

This specification describes the PCI Express® architecture, interconnect attributes, fabric management, and the programming interface required to design and build systems and peripherals that are compliant with the PCI Express Specification.

The goal is to enable such devices from different vendors to inter-operate in an open architecture. The specification is intended as an enhancement to the PCI™ architecture spanning multiple market segments; Clients (Desktops and Mobile), Servers (Standard and Enterprise), and Embedded and Communication devices. The specification allows system OEMs and peripheral developers adequate room for product versatility and market differentiation without the burden of carrying obsolete interfaces or losing compatibility.

Document Organization

The PCI Express Specification is organized as a base specification and a set of companion documents. The *PCI Express Base Specification* and the *PCI Express Card Electromechanical Specification* are among those that have been published. As the PCI Express definition evolves, other companion documents will be published.

The *PCI Express Base Specification* contains the technical details of the architecture, protocol, Link Layer, Physical Layer, and software interface. The *PCI Express Base Specification* is applicable to all variants of PCI Express.

The *PCI Express Card Electromechanical Specification* focuses on information necessary to implementing an evolutionary strategy with the PCI desktop/server mechanicals as well as electricals. The mechanical chapters of the specification contain a definition of evolutionary PCI Express card edge connectors while the electrical chapters cover auxiliary signals, power delivery, and the adapter interconnect electrical budget.

Documentation Conventions

Capitalization

Some terms are capitalized to distinguish their definition in the context of this document from their common English meaning. Words not capitalized have their common English meaning. When terms such as "memory write" or "memory read" appear completely in lower case, they include all transactions of that type.

Register names and the names of fields and bits in registers and headers are presented with the first letter capitalized and the remainder in lower case.

Numbers and Number Bases

Hexadecimal numbers are written with a lower case "h" suffix, e.g., FFFh and 80h. Hexadecimal numbers larger than four digits are represented with a space dividing each group of four digits, as in 1E FFFF FFFFh. Binary numbers are written with a lower case "b" suffix, e.g., 1001b and 10b.

Binary numbers larger than four digits are written with a space dividing each group of four digits, as in 1000 0101 0010b.

All other numbers are decimal.

Implementation Notes

Implementation Notes should not be considered to be part of this specification. They are included for clarification and illustration only.

Terms and Acronyms

8b/10b	The data encoding scheme[1] used in the PCI Express Physical Layer.
Access Control Services, ACS	A set of capabilities and control registers used to implement access control over routing within a PCI Express component.
ACS Violation	An error that applies to a Posted or Non-Posted Request when the Completer detects an access control violation.
adapter	Used generically to refer to an add-in card or module.
Advanced Error Reporting, AER	Advanced Error Reporting (see Section 7.10).
advertise (Credits)	Used in the context of Flow Control, the act of a Receiver sending information regarding its Flow Control Credit availability.
Alternative Routing-ID, ARI	Alternative Routing-ID Interpretation. Applicable to Requester IDs and Completer IDs as well as Routing IDs.
ARI Device	A ~~multi-Function~~ Device associated with an Upstream Port, ~~capable of supporting up to 256 Functions~~ whose Functions each contain an ARI Capability structure.
ARI Downstream Port	A Switch Downstream Port or Root Port that supports ARI Forwarding.
asserted	The active logical state of a conceptual or actual signal.
async removal	Removal of an adapter or cable from a slot without lock-step synchronization with the operating system (i.e., in an asynchronous manner without button presses, etc).
AtomicOp	One of three architected atomic operations where a single PCI Express transaction targeting a location in Memory Space reads the location's value, potentially writes a new value to the location, and returns the original value. This read-modify-write sequence to the location is performed atomically. AtomicOps include FetchAdd, Swap, and CAS.
attribute	Transaction handling preferences indicated by specified Packet header bits and fields (~~for example~~ e.g., non-snoop).
Beacon	An optional 30 kHz–500 MHz in-band signal used to exit the L2 Link power management state. One of two defined mechanisms for waking up a Link in L2 (see ~~also~~ wakeup).

Comment [VS1]: Errata A18

Comment [VS2]: ECN: Downstream Port containment (DPC)

[1] IBM Journal of Research and Development, Vol. 27, #5, September 1983 "A DC-Balanced, Partitioned-Block 8B/10B Transmission Code" by Widmer and Franaszek.

Bridge	One of several defined System Elements. A Function that virtually or actually connects a PCI/PCI-X segment or PCI Express Port with an internal component interconnect or with another PCI/PCI-X bus segment or PCI Express Port. A virtual *Bridge* in a Root Complex or Switch must use the software configuration interface described in this specification.
by-1, x1	A Link or Port with one Physical Lane.
by-8, x8	A Link or Port with eight Physical Lanes.
by-N, xN	A Link or Port with "N" Physical Lanes.
Character	An 8-bit quantity treated as an atomic entity; a byte.
Clear	A bit is Clear when its value is 0b.
cold reset	A Fundamental Reset following the application of power.
Compare and Swap, CAS	Compare and Swap. An AtomicOp where the value of a target location is compared to a specified value and, if they match, another specified value is written back to the location. Regardless, the original value of the location is returned.
Completer	The Function that terminates or "completes" a given Request, and generates a Completion if appropriate. Generally the Function targeted by the Request serves as the Completer. For cases when an uncorrectable error prevents the Request from reaching its targeted Function, the Function that detects and handles the error serves as the Completer.
Completer Abort, CA	1. A status that applies to a posted or non-posted Request that the Completer is permanently unable to complete successfully, due to a violation of the Completer's programming model or to an unrecoverable error associated with the Completer. 2. A status indication returned with a Completion for a non-posted Request that suffered a Completer Abort at the Completer.
Completer ID	The combination of a Completer's Bus Number, Device Number, and Function Number that uniquely identifies the Completer of the Request within a Hierarchy. With an ARI Completer ID, bits traditionally used for the Device Number field are used instead to expand the Function Number field, and the Device Number is implied to be 0.
Completion	A Packet used to terminate, or to partially terminate, a transaction sequence. A *Completion* always corresponds to a preceding Request, and, in some cases, includes data.
component	A physical device (a single package).
Configuration Space	One of the four address spaces within the PCI Express architecture. Packets with a *Configuration Space* address are used to configure ~~a~~ Function~~s within a device~~.
Configuration-Ready	A Function is "Configuration-Ready" when it is guaranteed that the Function will respond to a valid Configuration Request targeting the Function with a Completion indicating Successful Completion status.
conventional PCI	Behavior or features that conform to the *PCI Local Bus Specification.*
Conventional Reset	A Hot, Warm, or Cold reset. Distinct from Function Level Reset (FLR).
Data Link Layer	The intermediate Layer that is between the Transaction Layer and the Physical Layer.

Comment [VS3]: Errata A18Errata b29

Comment [VS4]: Errata A18

Comment [VS5]: D-ECN-Readiness Notifications

Data Link Layer Packet, DLLP	A Packet generated in the Data Link Layer to support Link management functions.
data payload	Information following the header in some packets that is destined for consumption by the targeted Function receiving the Packet (for example, Write Requests or Read Completions).
deasserted	The inactive logical state of a conceptual or actual signal.
Design for Testability, DFT	Design for Testability.
device	1. A physical or logical entity that performs a specific type of I/O. 2. A component on either end of a PCI Express Link. 3. A common imprecise synonym for Function, particularly when a device has a single Function.
Device	A collection of one or more Functions within a single Hierarchy identified by common Bus Number and Device Number. An SR-IOV Device may have additional Functions accessed via additional Bus Numbers configured through one or more SR-IOV Capability structures.
Device Readiness Status, DRS	A mechanism for indicating that a Device is Configuration-Ready (see Section 6.23.1)
Downstream	1. The relative position of an interconnect/System Element (Port/component) that is farther from the Root Complex. The Ports on a Switch that are not the Upstream Port are *Downstream* Ports. All Ports on a Root Complex are *Downstream* Ports. The *Downstream* component on a Link is the component farther from the Root Complex. 2. A direction of information flow where the information is flowing away from the Root Complex.
Downstream Port Containment, DPC	The automatic disabling of the Link below a Downstream Port following an uncorrectable error, which prevents TLPs subsequent to the error from propagating Upstream or Downstream.
DWORD, DW	Four bytes. Used in the context of a data payload, the 4 bytes of data must be on a naturally aligned 4-byte boundary (the least significant 2 bits of the byte address are 00b).
Egress Port	The transmitting Port; that is, the Port that sends outgoing traffic.
Electrical Idle	The state of the output driver in which both lines, D+ and D-, are driven to the DC common mode voltage.
End-End TLP Prefix	A TLP Prefix that is carried along with a TLP from source to destination. See Section 2.2.10.2.
Endpoint	One of several defined System Elements. A Function that has a Type 00h Configuration Space header.
error detection	Mechanisms that determine that an error exists, either by the first agent to discover the error (e.g., Malformed TLP) or by the recipient of a signaled error (e.g., receiver of a poisoned TLP).
error logging	A detector setting one or more bits in architected registers based on the detection of an error. The detector might be the original discoverer of an error or a recipient of a signaled error.

Comment [VS6]: ECN: Downstream Port containment (DPC)

error reporting	In a broad context, the general notification of errors. In the context of the Device Control register, sending an error Message. In the context of the Root Error Command register, signaling an interrupt as a result of receiving an error Message.
error signaling	One agent notifying another agent of an error either by (1) sending an error Message, (2) sending a Completion with UR/CA Status, or (3) poisoning a TLP.
Extended Function	Within an ARI Device, a Function whose Function Number is greater than 7. Extended Functions are accessible only after ARI-aware software has enabled ARI Forwarding in the Downstream Port immediately above the ARI Device.
FetchAdd	Fetch and Add. An AtomicOp where the value of a target location is incremented by a specified value using two's complement arithmetic ignoring any carry or overflow, and the result is written back to the location. The original value of the location is returned.
Flow Control	The method for communicating receive buffer status from a Receiver to a Transmitter to prevent receive buffer overflow and allow Transmitter compliance with ordering rules.
Flow Control Packet, FCP	A DLLP used to send Flow Control information from the Transaction Layer in one component to the Transaction Layer in another component.
Function	An Within a Device, an addressable entity in Configuration Space associated with a single Function Number. May be uUsed to refer to one Function of a multi-Function deviceDevice, or to the only Function in a single-Function deviceDevice. Specifically included are special types of Functions defined in the I/O Virtualization specifications, notably Physical Functions and Virtual Functions.
Function Group	Within an ARI Device, a configurable set of Functions that are associated with a single Function Group Number. Function Groups can optionally serve as the basis for virtual channel (VC) arbitration or access control between multiple Functions within the ARI Device.
Function Level Reset, FLR	A mechanism for resetting a specific Endpoint Function (see Section 6.6.2).
Function Readiness Status, FRS	A mechanism for indicating that a Function is Configuration-Ready (see Section 6.23.2).
Fundamental Reset	A hardware mechanism for setting or returning all Port states to the initial conditions specified in this document (see Section 6.6).
header	A set of fields that appear at or near the front of a Packet that contain the information required to determine the characteristics and purpose of the Packet.
Hierarchy	The A tree structured PCI Express I/O interconnect topology–, wherein the Configuration Space addresses (IDs) used for routing and Requester/Completer identification are unique. A system may contain multiple Hierarchies.
hierarchy domain	The part of a Hierarchy originating from a single Root Port.
Host Bridge	The pPart of a Root Complex that connects a host CPU or CPUs to a Hierarchy.
hot reset	A reset propagated in-band across a Link using a Physical Layer mechanism.
in-band signaling	A method for signaling events and conditions using the Link between two components, as opposed to the use of separate physical (sideband) signals. All mechanisms defined in this document can be implemented using *in-band signaling*, although in some form factors sideband signaling may be used instead.

Comment [VS7]: Errata A18

Comment [VS8]: D-ECN-Readiness Notifications

Comment [VS9]: Errata A19

Ingress Port	Receiving Port; that is, the Port that accepts incoming traffic.
Internal Error	An error associated with a PCI Express interface that occurs within a component and which may not be attributable to a packet or event on the PCI Express interface itself or on behalf of transactions initiated on PCI Express.
I/O Space	One of the four address spaces of the PCI Express architecture. Identical to the I/O Space defined in the *PCI Local Bus Specification*.
isochronous	Data associated with time-sensitive applications, such as audio or video applications.
invariant	A field of a TLP header or TLP Prefix that contains a value that cannot legally be modified as the TLP flows through the PCI Express fabric.
Lane	A set of differential signal pairs, one pair for transmission and one pair for reception. A by-N Link is composed of N *Lanes*.
Layer	A unit of distinction applied to this specification to help clarify the behavior of key elements. The use of the term *Layer* does not imply a specific implementation.
Link	The collection of two Ports and their interconnecting Lanes. A *Link* is a dual-simplex communications path between two components.
Lightweight Notification, LN	A lightweight protocol that supports notifications to Endpoints via a hardware mechanism when cachelines of interest are updated.
LN Completer, LNC	A service subsystem in the host that receives LN Read/Write Requests, and sends LN Messages when registered cachelines are updated.
LN Completion	A Completion whose TLP Header has the LN bit Set.
LN Message	An architected Message used for notifications with LN protocol.
LN Read	A Memory Read Request whose TLP Header has the LN bit Set.
LN Requester, LNR	A client subsystem in an Endpoint that sends LN Read/Write Requests and receives LN Messages.
LN Write	A Memory Write Request whose TLP Header has the LN bit Set.
Local Register Configuration, LRC	Local Register Configuration (LRC) is the process used by the Port to configure the parameters of the Local MODULE.
Local TLP Prefix	A TLP Prefix that is carried along with a TLP on a single Link. See Section 2.2.10.1.
Logical Bus	The logical connection among a collection of Devices that have the same Bus Number in Configuration Space.
Logical Idle	A period of one or more Symbol Times when no information (TLPs, DLLPs, or any special Symbol) is being transmitted or received. Unlike Electrical Idle, during *Logical Idle* the Idle data Symbol is being transmitted and received.
M-PHY	PHY defined by MIPI Alliance.
Malformed Packet	A TLP that violates specific TLP formation rules as defined in this specification.
Memory Space	One of the four address spaces of the PCI Express architecture. Identical to the Memory Space defined in PCI 3.0.

Comment [VS10]: ECN: Lightweight Notification (LN) Protocol

Comment [VS11]: ECN: M-PCIe

Comment [VS12]: ECN: M-PCIe

Message	A TLP used to communicate information outside of the Memory, I/O, and Configuration Spaces.
Message Signaled Interrupt (MSI/MSI-X)	Two similar but separate mechanisms that enable a Function to request service by writing a system-specified DWORD of data to a system-specified address using a Memory Write Request. Compared to MSI, MSI-X supports a larger maximum number of vectors and independent message address and data for each vector.
Message Space	One of the four address spaces of the PCI Express architecture.
Multicast, MC	A feature and associated mechanisms that enable a single Posted Request TLP sent by a source to be distributed to multiple targets.
Multicast Group, MCG	A set of Endpoints that are the target of Multicast TLPs in a particular address range.
Multicast Hit	The determination by a Receiver that a TLP will be handled as a Multicast TLP.
Multicast TLP	A TLP that is potentially distributed to multiple targets, as controlled by Multicast Capability structures in the components through which the TLP travels.
Multicast Window	A region of Memory Space where Posted Request TLPs that target it will be handled as Multicast TLPs.
Multi-Function Device, MFD	A Device that has multiple Functions.
naturally aligned	A data payload with a starting address equal to an integer multiple of a power of two, usually a specific power of two. For example, 64-byte *naturally aligned* means the least significant 6 bits of the byte address are 00 0000b.
P2P	Peer-to-peer.
PCI bus	The PCI Local Bus, as specified in the *PCI Local Bus Specification, Revision 3.0* and the *PCI-X Addendum to the PCI Local Bus Specification, Revision 2.0*.
PCI Software Model	The software model necessary to initialize, discover, configure, and use a PCI device, as specified in the *PCI Local Bus Specification, Revision 3.0*, the *PCI-X Addendum to the PCI Local Bus Specification, Revision 2.0*, and the *PCI BIOS Specification*.
Packet	A fundamental unit of information transfer consisting of an optional TLP Prefix, followed by a header and, in some cases, followed by a data payload.
Parts per Million, ppm	Parts per Million. Applied to frequency, the difference, in millionths of a Hertz, between a stated ideal frequency, and the measured long-term average of a frequency.
Phantom Function Number, PFN	An unclaimed Function Number that may be used to expand the number of outstanding transaction identifiers by logically combining the *PFN* with the Tag identifier to create a unique transaction identifier.
Physical Lane	See Lane.
Physical Layer	The Layer that directly interacts with the communication medium between two components.
Port	1. Logically, an interface between a component and a PCI Express Link. 2. Physically, a group of Transmitters and Receivers located on the same chip that define a Link.

Comment [VS13]: Errata A18

PMUX Channel	A multiplexed channel on a PMUX Link that is configured to transport a specific multiplexed protocol. See Appendix G.
PMUX Link	A Link where Protocol Multiplexing is supported and enabled. See Appendix G.
PMUX Packet	A non-PCI Express Packet transported over a PCI Express Link. See Appendix G.
Precision Time Measurement PTM	Optional capability for communicating precise timing information between components.
Process Address Space ID PASID	The Process Address Space ID, in conjunction with the Requester ID, uniquely identifies the address space associated with a transaction.
Quality of Service, QoS	Attributes affecting the bandwidth, latency, jitter, relative priority, etc., for differentiated classes of traffic.
QWORD, QW	Eight bytes. Used in the context of a data payload, the 8 bytes of data must be on a naturally aligned 8-byte boundary (the least significant 3 bits of the address are 000b).
Receiver, Rx	The component that receives Packet information across a Link.
Receiving Port	In the context of a specific TLP or DLLP, the Port that receives the Packet on a given Link.
Reported Error	An error subject to the logging and signaling requirements architecturally defined in this document.
Request	A Packet used to initiate a transaction sequence. A *Request* includes operation code and, in some cases, address and length, data, or other information.
Requester	The Function that first introduces a transaction sequence into the PCI Express domain.
Requester ID	The combination of a Requester's Bus Number, Device Number, and Function Number that uniquely identifies the Requester within a Hierarchy. With an ARI Requester ID, bits traditionally used for the Device Number field are used instead to expand the Function Number field, and the Device Number is implied to be 0.
Reserved	The contents, states, or information are not defined at this time. Using any *Reserved* area (for example, packet header bit-fields, configuration register bits) is not permitted. Reserved register fields must be read only and must return 0 (all 0's for multi-bit fields) when read. Reserved encodings for register and packet fields must not be used. Any implementation dependence on a *Reserved* field value or encoding will result in an implementation that is not PCI Express-compliant. The functionality of such an implementation cannot be guaranteed in this or any future revision of this specification.
Root Complex, RC	A defined System Element that includes ~~a~~ zero or more Host Bridges, zero or more Root Complex Integrated Endpoints, zero or more Root Complex Event Collectors, and one or more Root Ports.
Root Complex Component	A logical aggregation of Root Ports, Root Complex Register Blocks, ~~and~~ Root Complex Integrated Endpoints, and Root Complex Event Collectors.
Root Port	A PCI Express Port on a Root Complex that maps a portion of ~~a~~the Hierarchy through an associated virtual PCI-PCI Bridge.
RP PIO	Root Port Programmed I/O. See Section 6.2.10.3.

Comment [VS14]: ECN: Precision Time Measurement

Comment [VS15]: ECN:PASID

Comment [VS16]: Errata A18

Comment [VS17]: Errata A19

Comment [VS18]: A19

Comment [VS19]: A19

Comment [VS20]: ECN: Enhanced DPC

RRAP	Remote Register Access Protocol (RRAP) is the protocol used by the Upstream component to discover and configure the M-PHY MODULES of the Downstream component. RRAP is also used by the Test programmer to set up test modes.
RRC	Remote Register Configuration (RRC) is the process used by the Upstream component to access the capability and configuration attributes of the Downstream components M-PHY MODULES. The RRAP protocol must be used by the Upstream component to configure the Remote MODULE.
Set	A bit is Set when its value is 1b.
sideband signaling	A method for signaling events and conditions using physical signals separate from the signals forming the Link between two components. All mechanisms defined in this document can be implemented using in-band signaling, although in some form factors sideband signaling may be used instead.
single–Function Device SFD	A device that has a single Function.
slot	Used generically to refer to an add-in card slot, ~~or~~ module bay, or cable receptacle.
Split Transaction	A single logical transfer containing an initial transaction (the Request) terminated at the target (the Completer), followed by one or more transactions initiated by the Completer in response to the Request.
SR-IOV	*Single Root I/O Virtualization and Sharing*, a separate specification.
SR-IOV Device	A Device containing one or more Functions that have an SR-IOV Capability structure.
Switch	A defined System Element that connects two or more Ports to allow Packets to be routed from one Port to another. To configuration software, a *Switch* appears as a collection of virtual PCI-to-PCI Bridges.
Symbol	A 10-bit quantity when using 8b/10b encoding. An 8-bit quantity when using 128b/130b encoding.
Symbol Time	The period of time required to place a Symbol on a Lane (10 times the Unit Interval when using 8b/10b encoding and 8 times the Unit Interval when using 128b/130b encoding).
System Element	A defined Device or collection of devices that operate according to distinct sets of rules. The following *System Elements* are defined: Root Complex, Endpoint, Switch, and Bridge.
Tag	A number assigned to a given Non-posted Request to distinguish Completions for that Request from other Requests.
TLP Prefix	Additional information that may be optionally prepended to a TLP. TLP Prefixes are either Local or End-End. A TLP can have multiple TLP Prefixes. See Section 2.2.10.
Transaction Descriptor	An element of a Packet header that, in addition to Address, Length, and Type, describes the properties of the Transaction.
Transaction ID	A component of the Transaction Descriptor including Requester ID and Tag.
Transaction Layer	The Layer that operates at the level of transactions (for example, read, write).
Transaction Layer Packet, TLP	A Packet generated in the Transaction Layer to convey a Request or Completion.

Comment [VS21]: ECN: M-PCIe

Comment [VS22]: ECN: M-PCIe

Comment [VS23]: Errata A18

Comment [VS24]: ECN: Downstream Port containment (DPC)

Comment [VS25]: Errata A18

transaction sequence	A single Request and zero or more Completions associated with carrying out a single logical transfer by a Requester.
Transceiver	The physical Transmitter and Receiver pair on a single chip.
Transmitter, Tx	The component sending Packet information across a Link.
Transmitting Port	In the context of a specific TLP or DLLP, the Port that transmits the Packet on a given Link.
Unconditional Swap, Swap	Unconditional Swap. An AtomicOp where a specified value is written to a target location, and the original value of the location is returned.
Unit Interval, UI	Given a data stream of a repeating pattern of alternating 1 and 0 values, the *Unit Interval* is the value measured by averaging the time interval between voltage transitions, over a time interval long enough to make all intentional frequency modulation of the source clock negligible.
Unsupported Request, Upstream	1. The relative position of an interconnect/System Element (Port/component) that is closer to the Root Complex. The Port on a Switch that is closest topologically to the Root Complex is the *Upstream* Port. The Port on a component that contains only Endpoint or Bridge Functions is an *Upstream* Port. The *Upstream* component on a Link is the component closer to the Root Complex. 2. A direction of information flow where the information is flowing towards the Root Complex.
UR	1. A status that applies to a posted or non-posted Request that specifies some action or access to some space that is not supported by the Completer. 2. A status indication returned with a Completion for a non-posted Request that suffered an Unsupported Request at the Completer.
variant	A field of a TLP header ~~or TLP Prefix~~ that contains a value that is subject to possible modification according to the rules of this specification as the TLP flows through the PCI Express fabric.
wakeup	An optional mechanism used by a component to request the reapplication of main power when in the L2 Link state. Two such mechanisms are defined: Beacon (using in-band signaling) and WAKE# (using sideband signaling).
warm reset	A Fundamental Reset without cycling the supplied power.

Comment [VS26]: Errata B8

Reference Documents

PCI Express Card Electromechanical Specification, Revision ~~3~~2.0

PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0

PCI Express Mini Card Electromechanical Specification, Revision 1.2

PCI Local Bus Specification, Revision 3.0

PCI-X Addendum to the PCI Local Bus Specification, Revision 2.0

PCI Hot-Plug Specification, Revision 1.1

PCI Standard Hot-Plug Controller and Subsystem Specification, Revision 1.0

PCI-to-PCI Bridge Architecture Specification, Revision 1.2

PCI Bus Power Management Interface Specification, Revision 1.2

PCI Firmware Specification, Revision 3.0

Address Translation Services, Revision 1.1

Advanced Configuration and Power Interface Specification, Revision 3.0b

Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority

MIPI Alliance Specification for M-PHY version 2.0 – 4 April, 2012, see http://mipi.org

Comment [VS27]: ECN: M-PCIe

1

1. Introduction

This chapter presents an overview of the PCI Express architecture and key concepts. PCI Express is a high performance, general purpose I/O interconnect defined for a wide variety of future computing and communication platforms. Key PCI attributes, such as its usage model, load-store architecture, and software interfaces, are maintained, whereas its parallel bus implementation is replaced by a highly scalable, fully serial interface. PCI Express takes advantage of recent advances in point-to-point interconnects, Switch-based technology, and packetized protocol to deliver new levels of performance and features. Power Management, Quality Of Service (QoS), Hot-Plug/Hot-Swap support, Data Integrity, and Error Handling are among some of the advanced features supported by PCI Express.

Comment [VS28]: ECN: Downstream Port containment (DPC)

1.1. A Third Generation I/O Interconnect

The high-level requirements for this third generation I/O interconnect are as follows:

- **Supports multiple market segments and emerging applications:**
 - Unifying I/O architecture for desktop, mobile, workstation, server, communications platforms, and embedded devices
- **Ability to deliver low cost, high volume solutions:**
 - Cost at or below PCI cost structure at the system level
- **Support multiple platform interconnect usages:**
 - Chip-to-chip, board-to-board via connector or cabling
- **New mechanical form factors:**
 - Mobile, PCI-like form factor and modular, cartridge form factor
- **PCI compatible software model:**
 - Ability to enumerate and configure PCI Express hardware using PCI system configuration software implementations with no modifications
 - Ability to boot existing operating systems with no modifications
 - Ability to support existing I/O device drivers with no modifications
 - Ability to configure/enable new PCI Express functionality by adopting the PCI configuration paradigm

- **Performance:**
 - Low-overhead, low-latency communications to maximize application payload bandwidth and Link efficiency
 - High-bandwidth per pin to minimize pin count per device and connector interface
 - Scalable performance via aggregated Lanes and signaling frequency
- **Advanced features:**
 - Comprehend different data types and ordering rules
 - Power management and budgeting
 - Ability to identify power management capabilities of a given Function
 - Ability to transition a Function into a specific power state
 - Ability to receive notification of the current power state of a Function
 - Ability to generate a request to wakeup from a power-off state of the main power supply
 - Ability to sequence device power-up to allow graceful platform policy in power budgeting.
 - M-PCIe
 - Ability to operate PCIe Data Link and Transaction Layer protocol over MIPI M-PHY
 - Ability to maintain compatibility with PCI Express programming models
 - Ability to support Multi-lane configurations
 - Ability to support Asymmetric link width configurations
 - Ability to support Dynamic bandwidth scalability
 - Ability to optimize the interconnect for RFI/EMI
 - Ability to enable short channel circuit optimizations
 - Provide scalability for all MIPI M-PHY high speed gears
 - Ability to support M-PHY TYPE I MODULE only
 - Ability to support for MIPI M-PHY LS gear to be utilized for M-PHY parameter initialization.
 - Ability to support shared and independent reference clocks
 - Ability to support differentiated services, i.e., different qualities of service (QoS)
 - Ability to have dedicated Link resources per QoS data flow to improve fabric efficiency and effective application-level performance in the face of head-of-line blocking
 - Ability to configure fabric QoS arbitration policies within every component
 - Ability to tag end-to-end QoS with each packet

Comment [VS29]: ECN: M-PCIe

- Ability to create end-to-end isochronous (time-based, injection rate control) solutions

- Hot-Plug ~~and Hot-Swap~~ support
 - Ability to support existing PCI Hot-Plug ~~and Hot-Swap~~ solutions
 - Ability to support native Hot-Plug ~~and Hot-Swap~~ solutions (no sideband signals required)
 - Ability to support async removal
 - Ability to support a unified software model for all form factors
- Data Integrity
 - Ability to support Link-level data integrity for all types of transaction and Data Link packets
 - Ability to support end-to-end data integrity for high availability solutions
- Error Handling
 - Ability to support PCI-level error handling
 - Ability to support advanced error reporting and handling to improve fault isolation and recovery solutions
- Process Technology Independence
 - Ability to support different DC common mode voltages at Transmitter and Receiver
- Ease of Testing
 - Ability to test electrical compliance via simple connection to test equipment

Comment [VS30]: ECN: Downstream Port containment (DPC)

Comment [VS31]: ECN: Downstream Port containment (DPC)

1.2. PCI Express Link

A Link represents a dual-simplex communications channel between two components. The fundamental PCI Express Link consists of two, low-voltage, differentially driven signal pairs: a Transmit pair and a Receive pair as shown in Figure 1-1. A PCI Express Link consists of a PCIe PHY as defined in Chapter 4 whereas the M-PCIe Link consists of M-PHY, a physical layer specification developed by MIPI Alliance.

Comment [VS32]: ECN: M-PCIe

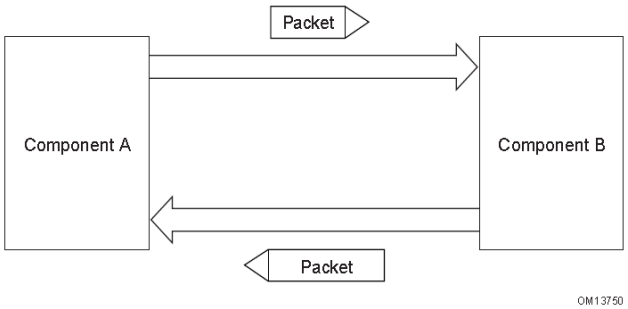


Figure 1-1: PCI Express Link

The primary Link attributes for PCI Express Link are:

Comment [VS33]: ECN: M-PCIe

- The basic Link – PCI Express Link consists of dual unidirectional differential Links, implemented as a Transmit pair and a Receive pair. A data clock is embedded using an encoding scheme (see Chapter 4) to achieve very high data rates.
- Signaling rate – Once initialized, each Link must only operate at one of the supported signaling levels. For the first generation of PCI Express technology, there is only one signaling rate defined, which provides an effective 2.5 Gigabits/second/Lane/direction of raw bandwidth. The second generation provides an effective 5.0 Gigabits/second/Lane/direction of raw bandwidth. The third generation provides an effective 8.0 Gigabits/second/Lane/direction of raw bandwidth. The data rate is expected to increase with technology advances in the future.
- Lanes – A Link must support at least one Lane – each Lane represents a set of differential signal pairs (one pair for transmission, one pair for reception). To scale bandwidth, a Link may aggregate multiple Lanes denoted by xN where N may be any of the supported Link widths. A x8 Link operating at the 2.5 GT/s data rate represents an aggregate bandwidth of 20 Gigabits/second of raw bandwidth in each direction. This specification describes operations for x1, x2, x4, x8, x12, x16, and x32 Lane widths.
- Initialization – During hardware initialization, each PCI Express Link is set up following a negotiation of Lane widths and frequency of operation by the two agents at each end of the Link. No firmware or operating system software is involved.
- Symmetry – Each Link must support a symmetric number of Lanes in each direction, i.e., a x16 Link indicates there are 16 differential signal pairs in each direction.

The primary Link attributes for M-PCIe Link are defined in Chapter 8.

Comment [VS34]: ECN: M-PCIe

1.3. PCI Express Fabric Topology

A fabric is composed of point-to-point Links that interconnect a set of components – an example fabric topology is shown in Figure 1-2. This figure illustrates a single fabric instance referred to as a hierarchy – composed of a Root Complex (RC), multiple Endpoints (I/O devices), a Switch, and a PCI Express to PCI/PCI-X Bridge, all interconnected via PCI Express Links.

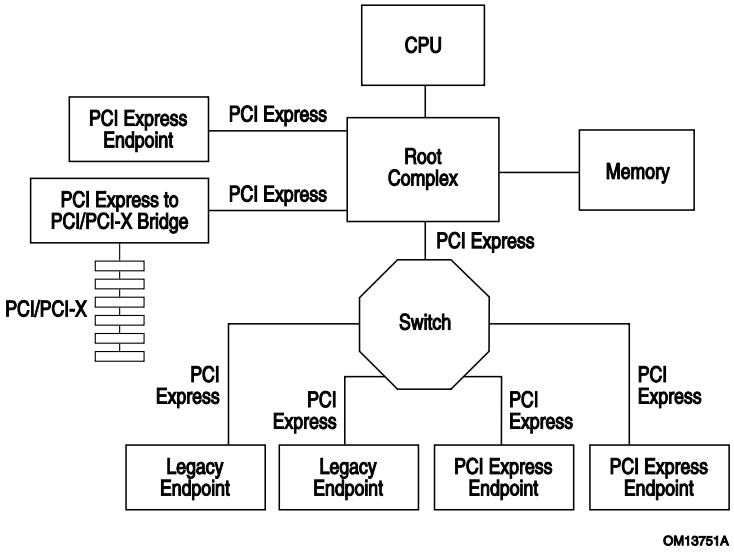


Figure 1-2: Example Topology

1.3.1. Root Complex

- A Root Complex (RC) denotes the root of an I/O hierarchy that connects the CPU/memory subsystem to the I/O.
- As illustrated in Figure 1-2, a Root Complex may support one or more PCI Express Ports. Each interface defines a separate hierarchy domain. Each hierarchy domain may be composed of a single Endpoint or a sub-hierarchy containing one or more Switch components and Endpoints.
- The capability to route peer-to-peer transactions between hierarchy domains through a Root Complex is optional and implementation dependent. For example, an implementation may incorporate a real or virtual Switch internally within the Root Complex to enable full peer-to-peer support in a software transparent way.

 Unlike the rules for a Switch, a Root Complex is generally permitted to split a packet into smaller packets when routing transactions peer-to-peer between hierarchy domains (except as noted below), e.g., split a single packet with a 256-byte payload into two packets of 128 bytes payload each. The resulting packets are subject to the normal packet formation rules contained in this specification (e.g., Max_Payload_Size, Read Completion Boundary, etc.). Component designers should note that splitting a packet into smaller packets may have negative performance

consequences, especially for a transaction addressing a device behind a PCI Express to PCI/PCI-X bridge.

Exception: A Root Complex that supports peer-to-peer routing of Vendor_Defined Messages is not permitted to split a Vendor_Defined Message packet into smaller packets except at 128-byte boundaries (i.e., all resulting packets except the last must be an integral multiple of 128 bytes in length) in order to retain the ability to forward the Message across a PCI Express to PCI/PCI-X Bridge. Refer to the *PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0* for additional information.

- A Root Complex must support generation of configuration requests as a Requester.
- A Root Complex is permitted to support the generation of I/O Requests as a Requester.
 - A Root Complex is permitted to generate I/O Requests to either or both of locations 80h and 84h to a selected Root Port, without regard to that Root Port's PCI Bridge I/O decode configuration; it is recommended that this mechanism only be enabled when specifically needed.
- A Root Complex must not support Lock semantics as a Completer.
- A Root Complex is permitted to support generation of Locked Requests as a Requester.

1.3.2. Endpoints

Endpoint refers to a type of Function that can be the Requester or Completer of a PCI Express transaction either on its own behalf or on behalf of a distinct non-PCI Express device (other than a PCI device or Host CPU), e.g., a PCI Express attached graphics controller or a PCI Express-USB host controller. Endpoints are classified as either legacy, PCI Express, or Root Complex Integrated Endpoints.

1.3.2.1. Legacy Endpoint Rules

- A Legacy Endpoint must be a Function with a Type 00h Configuration Space header.
- A Legacy Endpoint must support Configuration Requests as a Completer.
- A Legacy Endpoint may support I/O Requests as a Completer.
 - A Legacy Endpoint is permitted to accept I/O Requests to either or both of locations 80h and 84h, without regard to that Endpoint's I/O decode configuration.
- A Legacy Endpoint may generate I/O Requests.
- A Legacy Endpoint may support Lock memory semantics as a Completer if that is required by the device's legacy software support requirements.
- A Legacy Endpoint must not issue a Locked Request.
- A Legacy Endpoint may implement Extended Configuration Space Capabilities, but such Capabilities may be ignored by software.
- A Legacy Endpoint operating as the Requester of a Memory Transaction is not required to be capable of generating addresses 4 GB or greater.

- A Legacy Endpoint is required to support MSI or MSI-X or both if an interrupt resource is requested. If MSI is implemented, a Legacy Endpoint is permitted to support either the 32-bit or 64-bit Message Address version of the MSI Capability structure.
- A Legacy Endpoint is permitted to support 32-bit addressing for Base Address registers that request memory resources.
- A Legacy Endpoint must appear within one of the hierarchy domains originated by the Root Complex.

1.3.2.2. PCI Express Endpoint Rules

- A PCI Express Endpoint must be a Function with a Type 00h Configuration Space header.
- A PCI Express Endpoint must support Configuration Requests as a Completer.
- A PCI Express Endpoint must not depend on operating system allocation of I/O resources claimed through BAR(s).
- A PCI Express Endpoint must not generate I/O Requests.
- A PCI Express Endpoint must not support Locked Requests as a Completer or generate them as a Requester. PCI Express-compliant software drivers and applications must be written to prevent the use of lock semantics when accessing a PCI Express Endpoint.
- A PCI Express Endpoint operating as the Requester of a Memory Transaction is required to be capable of generating addresses greater than 4 GB.
- A PCI Express Endpoint is required to support MSI or MSI-X or both if an interrupt resource is requested., If MSI is implemented, a PCI Express Endpoint must support the 64-bit Message Address version of the MSI Capability structure.
- A PCI Express Endpoint requesting memory resources through a BAR must set the BAR's Prefetchable bit unless the range contains locations with read side-effects or locations in which the Function does not tolerate write merging. See Section 7.5.2.1 for additional guidance on having the Prefetchable bit Set.
- For a PCI Express Endpoint, 64-bit addressing must be supported for all BARs that have the prefetchable bit set. 32-bit addressing is permitted for all BARs that do not have the prefetchable bit set.
- The minimum memory address range requested by a BAR is 128 bytes.
- A PCI Express Endpoint must appear within one of the hierarchy domains originated by the Root Complex.

1.3.2.3. Root Complex Integrated Endpoint Rules

- A Root Complex Integrated Endpoint is implemented on internal logic of Root Complexes that contains the Root Ports.
- A Root Complex Integrated Endpoint must be a Function with a Type 00h Configuration Space header.
- A Root Complex Integrated Endpoint must support Configuration Requests as a Completer.
- A Root Complex Integrated Endpoint must not require I/O resources claimed through BAR(s).
- A Root Complex Integrated Endpoint must not generate I/O Requests.
- A Root Complex Integrated Endpoint must not support Locked Requests as a Completer or generate them as a Requester. PCI Express-compliant software drivers and applications must be written to prevent the use of lock semantics when accessing a Root Complex Integrated Endpoint.
- A Root Complex Integrated Endpoint operating as the Requester of a Memory Transaction is required to be capable of generating addresses equal to or greater than the Host is capable of handling as a Completer.
- A Root Complex Integrated Endpoint is required to support MSI or MSI-X or both if an interrupt resource is requested. If MSI is implemented, a Root Complex Integrated Endpoint is permitted to support either the 32-bit or 64-bit Message Address version of the MSI Capability structure.
- A Root Complex Integrated Endpoint is permitted to support 32-bit addressing for Base Address registers that request memory resources.
- A Root Complex Integrated Endpoint must not implement Link Capabilities, Link Status, Link Control, Link Capabilities 2, Link Status 2, and Link Control 2 registers in the PCI Express Extended Capability. See Section 7.8 for more details.
- A Root Complex Integrated Endpoint must signal PME and error conditions through the same mechanisms used on PCI systems. If a Root Complex Event Collector is implemented, a Root Complex Integrated Endpoint may optionally signal PME and error conditions through a Root Complex Event Collector. In this case, a Root Complex Integrated Endpoint must be associated with no more than one Root Complex Event Collector.
- A Root Complex Integrated Endpoint does not implement Active State Power Management.
- A Root Complex Integrated Endpoint may not be hot-plugged independent of the Root Complex as a whole.
- A Root Complex Integrated Endpoint must not appear in any of the hierarchy domains exposed by the Root Complex.
- A Root Complex Integrated Endpoint must not appear in Switches.

1.3.3. Switch

A Switch is defined as a logical assembly of multiple virtual PCI-to-PCI Bridge devices as illustrated in Figure 1-3. All Switches are governed by the following base rules.

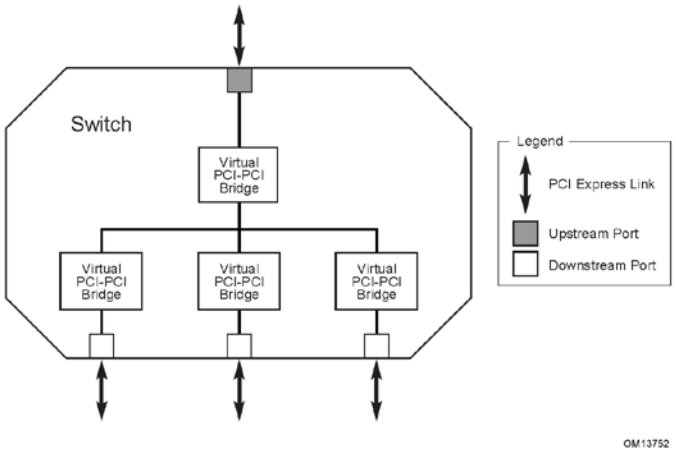


Figure 1-3: Logical Block Diagram of a Switch

- Switches appear to configuration software as two or more logical PCI-to-PCI Bridges.
- A Switch forwards transactions using PCI Bridge mechanisms; e.g., address based routing except when engaged in a Multicast, as defined in Section 6.14.
- Except as noted in this document, a Switch must forward all types of Transaction Layer Packets between any set of Ports.
- Locked Requests must be supported as specified in Section 6.5. Switches are not required to support Downstream Ports as initiating Ports for Locked requests.
- Each enabled Switch Port must comply with the flow control specification within this document.
- A Switch is not allowed to split a packet into smaller packets, e.g., a single packet with a 256-byte payload must not be divided into two packets of 128 bytes payload each.
- Arbitration between Ingress Ports (inbound Link) of a Switch may be implemented using round robin or weighted round robin when contention occurs on the same Virtual Channel. This is described in more detail later within the specification.
- Endpoints (represented by Type 00h Configuration Space headers) must not appear to configuration software on the Switch's internal bus as peers of the virtual PCI-to-PCI Bridges representing the Switch Downstream Ports.

1.3.4. Root Complex Event Collector

- A Root Complex Event Collector provides support for terminating error and PME messages from Root Complex Integrated Endpoints.
- A Root Complex Event Collector must follow all rules for a Root Complex Integrated Endpoint.
- A Root Complex Event Collector is not required to decode any memory or IO resources.
- A Root Complex Event Collector is identified by its Device/Port Type value (see section 7.8.2).
- A Root Complex Event Collector has the Base Class 08h, Sub-Class 076h and Programming Interface 00h.[2]
- A Root Complex Event Collector resides on the same Logical Bus as the Root Complex Integrated Endpoints it supports.
- Multiple Root Complex Event Collectors are permitted to reside on a single Logical Bus.
- A Root Complex Event Collector explicitly declares supported Root Complex Integrated Endpoints through the Root Complex Event Collector Endpoint Association Capability.
- Root Complex Event Collectors are optional.

Comment [VS35]: ECN: Change Root Complex Event Collector Class Code

Comment [VS36]: ECN: Change Root Complex Event Collector Class Code

1.3.5. PCI Express to PCI/PCI-X Bridge

- A PCI Express to PCI/PCI-X Bridge provides a connection between a PCI Express fabric and a PCI/PCI-X hierarchy.
- PCI Express Port(s) of a PCI Express to PCI/PCI-X Bridge must comply with the requirements of this document.

1.4. PCI Express Fabric Topology Configuration

The PCI Express Configuration model supports two mechanisms:

- PCI compatible configuration mechanism: The PCI compatible mechanism supports 100% binary compatibility with PCI 3.0 or later operating systems and their corresponding bus enumeration and configuration software.
- PCI Express enhanced configuration mechanism: The enhanced mechanism is provided to increase the size of available Configuration Space and to optimize access mechanisms.

Each PCI Express Link is mapped through a virtual PCI-to-PCI Bridge structure and has a logical PCI bus associated with it. The virtual PCI-to-PCI Bridge structure may be part of a PCI Express Root Complex Port, a Switch Upstream Port, or a Switch Downstream Port. A Root Port is a virtual PCI-to-PCI Bridge structure that originates a PCI Express hierarchy domain from a PCI

[2] Since an earlier version of this specification used Sub-Class 06h for this purpose, an implementation is still permitted to use Sub-Class 06h, but this is strongly discouraged.

Express Root Complex. Devices are mapped into Configuration Space such that each will respond to a particular Device Number.

1.5. PCI Express Layering Overview

This document specifies the architecture in terms of three discrete logical layers: the Transaction Layer, the Data Link Layer, and the Physical Layer. Each of these layers is divided into two sections: one that processes outbound (to be transmitted) information and one that processes inbound (received) information, as shown in Figure 1-4.

The fundamental goal of this layering definition is to facilitate the reader's understanding of the specification. Note that this layering does not imply a particular PCI Express implementation.

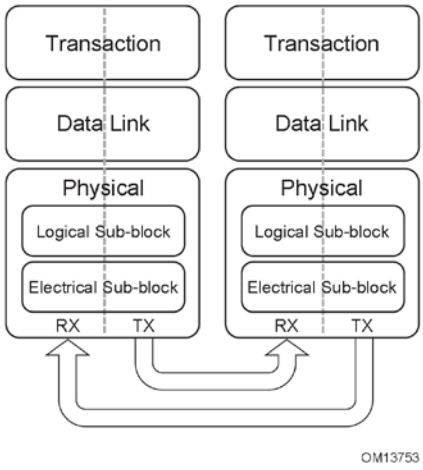


Figure 1-4: High-Level Layering Diagram

PCI Express uses packets to communicate information between components. Packets are formed in the Transaction and Data Link Layers to carry the information from the transmitting component to the receiving component. As the transmitted packets flow through the other layers, they are extended with additional information necessary to handle packets at those layers. At the receiving side the reverse process occurs and packets get transformed from their Physical Layer representation to the Data Link Layer representation and finally (for Transaction Layer Packets) to the form that can be processed by the Transaction Layer of the receiving device. Figure 1-5 shows the conceptual flow of transaction level packet information through the layers.

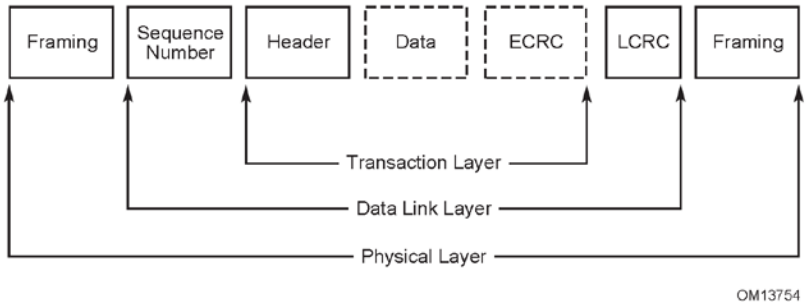


Figure 1-5: Packet Flow Through the Layers

Note that a simpler form of packet communication is supported between two Data Link Layers (connected to the same Link) for the purpose of Link management.

1.5.1. Transaction Layer

The upper Layer of the architecture is the Transaction Layer. The Transaction Layer's primary responsibility is the assembly and disassembly of Transaction Layer Packets (TLPs). TLPs are used to communicate transactions, such as read and write, as well as certain types of events. The Transaction Layer is also responsible for managing credit-based flow control for TLPs.

Every request packet requiring a response packet is implemented as a split transaction. Each packet has a unique identifier that enables response packets to be directed to the correct originator. The packet format supports different forms of addressing depending on the type of the transaction (Memory, I/O, Configuration, and Message). The Packets may also have attributes such as No Snoop, Relaxed Ordering, and ID-Based Ordering (IDO).

The transaction Layer supports four address spaces: it includes the three PCI address spaces (memory, I/O, and configuration) and adds Message Space. This specification uses Message Space to support all prior sideband signals, such as interrupts, power-management requests, and so on, as in-band Message transactions. You could think of PCI Express Message transactions as "virtual wires" since their effect is to eliminate the wide array of sideband signals currently used in a platform implementation.

1.5.2. Data Link Layer

The middle Layer in the stack, the Data Link Layer, serves as an intermediate stage between the Transaction Layer and the Physical Layer. The primary responsibilities of the Data Link Layer include Link management and data integrity, including error detection and error correction.

The transmission side of the Data Link Layer accepts TLPs assembled by the Transaction Layer, calculates and applies a data protection code and TLP sequence number, and submits them to Physical Layer for transmission across the Link. The receiving Data Link Layer is responsible for checking the integrity of received TLPs and for submitting them to the Transaction Layer for further processing. On detection of TLP error(s), this Layer is responsible for requesting

retransmission of TLPs until information is correctly received, or the Link is determined to have failed.

The Data Link Layer also generates and consumes packets that are used for Link management functions. To differentiate these packets from those used by the Transaction Layer (TLP), the term Data Link Layer Packet (DLLP) will be used when referring to packets that are generated and consumed at the Data Link Layer.

1.5.3. Physical Layer

The Physical Layer includes all circuitry for interface operation, including driver and input buffers, parallel-to-serial and serial-to-parallel conversion, PLL(s), and impedance matching circuitry. It includes also logical functions related to interface initialization and maintenance. The Physical Layer exchanges information with the Data Link Layer in an implementation-specific format. This Layer is responsible for converting information received from the Data Link Layer into an appropriate serialized format and transmitting it across the PCI Express Link at a frequency and width compatible with the device connected to the other side of the Link.

There are two Physical Layer implementations supported by this specification, the PCI Express Link consists of a PCIe Logical Physical and Electrical Physical Layer as defined in Chapter 4 whereas the M-PCIe Link consists of a Logical Physical Layer as defined in Chapter 8 and the Electrical Physical Layer as defined by M-PHY specification developed by MIPI Alliance.

Comment [VS37]: ECN: M-PCIe

The PCI Express architecture has "hooks" to support future performance enhancements via speed upgrades and advanced encoding techniques. The future speeds, encoding techniques or media may only impact the Physical Layer definition.

1.5.4. Layer Functions and Services

1.5.4.1. Transaction Layer Services

The Transaction Layer, in the process of generating and receiving TLPs, exchanges Flow Control information with its complementary Transaction Layer on the other side of the Link. It is also responsible for supporting both software and hardware-initiated power management.

Initialization and configuration functions require the Transaction Layer to:

- Store Link configuration information generated by the processor or management device
- Store Link capabilities generated by Physical Layer hardware negotiation of width and operational frequency

A Transaction Layer's Packet generation and processing services require it to:

- Generate TLPs from device core Requests
- Convert received Request TLPs into Requests for the device core
- Convert received Completion Packets into a payload, or status information, deliverable to the core
- Detect unsupported TLPs and invoke appropriate mechanisms for handling them

- If end-to-end data integrity is supported, generate the end-to-end data integrity CRC and update the TLP header accordingly. Flow control services:
- The Transaction Layer tracks flow control credits for TLPs across the Link.
- Transaction credit status is periodically transmitted to the remote Transaction Layer using transport services of the Data Link Layer.
- Remote Flow Control information is used to throttle TLP transmission.

Ordering rules:

- PCI/PCI-X compliant producer consumer ordering model
- Extensions to support Relaxed Ordering
- Extensions to support ID-Based Ordering

Power management services:

- ACPI/PCI power management, as dictated by system software.
- Hardware-controlled autonomous power management minimizes power during full-on power states.

Virtual Channels and Traffic Class:

- The combination of Virtual Channel mechanism and Traffic Class identification is provided to support differentiated services and QoS support for certain classes of applications.
- Virtual Channels: Virtual Channels provide a means to support multiple independent logical data flows over given common physical resources of the Link. Conceptually this involves multiplexing different data flows onto a single physical Link.
- Traffic Class: The Traffic Class is a Transaction Layer Packet label that is transmitted unmodified end-to-end through the fabric. At every service point (e.g., Switch) within the fabric, Traffic Class labels are used to apply appropriate servicing policies. Each Traffic Class label defines a unique ordering domain - no ordering guarantees are provided for packets that contain different Traffic Class labels.

1.5.4.2. Data Link Layer Services

The Data Link Layer is responsible for reliably exchanging information with its counterpart on the opposite side of the Link.

Initialization and power management services:

- Accept power state Requests from the Transaction Layer and convey to the Physical Layer
- Convey active/reset/disconnected/power managed state to the Transaction Layer

Data protection, error checking, and retry services:

- CRC generation
- Transmitted TLP storage for Data Link level retry
- Error checking
- TLP acknowledgment and retry Messages
- Error indication for error reporting and logging

1.5.4.3. Physical Layer Services

Interface initialization, maintenance control, and status tracking:

- Reset/Hot-Plug control/status
- Interconnect power management
- Width and Lane mapping negotiation
- Lane polarity inversion

Symbol and special Ordered Set generation:

- 8b/10b encoding/decoding
- Embedded clock tuning and alignment

Symbol transmission and alignment:

- Transmission circuits
- Reception circuits
- Elastic buffer at receiving side
- Multi-Lane de-skew (for widths > x1) at receiving side

System DFT support features

1.5.4.4. Inter-Layer Interfaces

1.5.4.4.1. Transaction/Data Link Interface

The Transaction to Data Link interface provides:

- Byte or multi-byte data to be sent across the Link
 - Local TLP-transfer handshake mechanism
 - TLP boundary information
- Requested power state for the Link

The Data Link to Transaction interface provides:

- Byte or multi-byte data received from the PCI Express Link
- TLP framing information for the received byte
- Actual power state for the Link
- Link status information

1.5.4.4.2. Data Link/Physical Interface

The Data Link to Physical interface provides:

- Byte or multi-byte wide data to be sent across the Link
 - Data transfer handshake mechanism
 - TLP and DLLP boundary information for bytes
- Requested power state for the Link

The Physical to Data Link interface provides:

- Byte or multi-byte wide data received from the PCI Express Link
- TLP and DLLP framing information for data
- Indication of errors detected by the Physical Layer
- Actual power state for the Link
- Connection status information

2

2. Transaction Layer Specification

2.1. Transaction Layer Overview

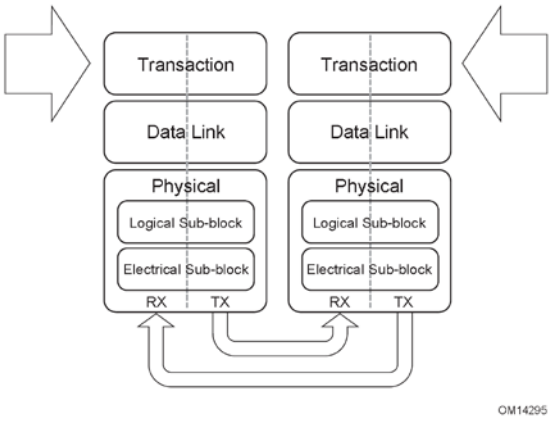


Figure 2-1: Layering Diagram Highlighting the Transaction Layer

At a high level, the key aspects of the Transaction Layer are:

- A pipelined full split-transaction protocol
- Mechanisms for differentiating the ordering and processing requirements of Transaction Layer Packets (TLPs)
- Credit-based flow control
- Optional support for data poisoning and end-to-end data integrity detection.

The Transaction Layer comprehends the following:

- TLP construction and processing
- Association of transaction-level mechanisms with device resources including:
 - Flow Control
 - Virtual Channel management
- Rules for ordering and management of TLPs
 - PCI/PCI-X compatible ordering
 - Including Traffic Class differentiation

This chapter specifies the behaviors associated with the Transaction Layer.

2.1.1. Address Spaces, Transaction Types, and Usage

Transactions form the basis for information transfer between a Requester and Completer. Four address spaces are defined, and different Transaction types are defined, each with its own unique intended usage, as shown in Table 2-1.

Table 2-1: Transaction Types for Different Address Spaces

Address Space	Transaction Types	Basic Usage
Memory	Read Write	Transfer data to/from a memory-mapped location
I/O	Read Write	Transfer data to/from an I/O-mapped location
Configuration	Read Write	Device Function configuration/setup
Message	Baseline (including Vendor–defined)	From event signaling mechanism to general purpose messaging

Details about the rules associated with usage of these address formats and the associated TLP formats are described later in this chapter.

2.1.1.1. Memory Transactions

Memory Transactions include the following types:

- Read Request/Completion
- Write Request
- AtomicOp Request/Completion

Memory Transactions use two different address formats:

- Short Address Format: 32-bit address
- Long Address Format: 64-bit address

Certain Memory Transactions can optionally have a PASID TLP Prefix containing the Process Address Space ID (PASID). See Section 6.20 for details.

Comment [VS38]: ECN: PASID

2.1.1.2. I/O Transactions

PCI Express supports I/O Space for compatibility with legacy devices which require their use. Future revisions of this specification may deprecate the use of I/O Space. I/O Transactions include the following types:

- Read Request/Completion
- Write Request/Completion

I/O Transactions use a single address format:

- Short Address Format: 32-bit address

2.1.1.3. Configuration Transactions

Configuration Transactions are used to access configuration registers of Functions within devices.

Configuration Transactions include the following types:

- Read Request/Completion
- Write Request/Completion

2.1.1.4. Message Transactions

The Message Transactions, or simply Messages, are used to support in-band communication of events between devices.

In addition to ~~the~~ specific~~ed~~ Messages defined in this document, PCI Express provides support for vendor-defined Messages using specified Message codes. Except for Vendor-Defined Messages that use the PCI-SIG® Vendor ID (0001h), the ~~The~~ definition of specific vendor-defined Messages is outside the scope of this document.

Comment [VS39]: ECN: Lightweight Notification (LP) Protocol

This specification establishes a standard framework within which vendors can specify their own vendor-defined Messages tailored to fit the specific requirements of their platforms (see Sections 2.2.8.5 and 2.2.8.7).

Note that these vendor-defined Messages are not guaranteed to be interoperable with components from different vendors.

2.1.2. Packet Format Overview

Transactions consist of Requests and Completions, which are communicated using packets. Figure 2-2 shows a high level serialized view of a Transaction Layer Packet (TLP), consisting of one or more optional TLP Prefixes, a TLP header, a data payload (for some types of packets), and an optional TLP digest. Figure 2-3 shows a more detailed view of the TLP. The following sections of this chapter define the detailed structure of the packet headers and digest.

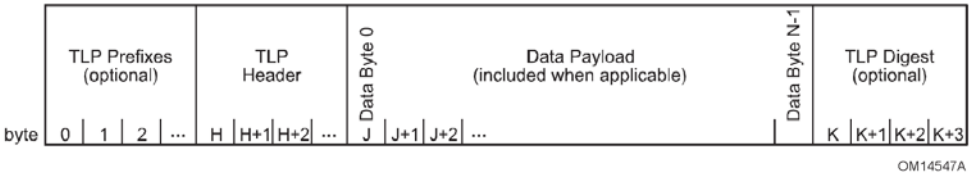


Figure 2-2: Serial View of a TLP

PCI Express conceptually transfers information as a serialized stream of bytes as shown in Figure 2-2. Note that at the byte level, information is transmitted/received over the interconnect with the leftmost byte of the TLP as shown in Figure 2-2 being transmitted/received first (byte 0 if one or more optional TLP Prefixes are present else byte H). Refer to Section 4.2 for details on how individual bytes of the packet are encoded and transmitted over the physical media.

Detailed layouts of the TLP Prefix, TLP Header and TLP Digest (presented in generic form in Figure 2-3) are drawn with the lower numbered bytes on the left rather than on the right as has traditionally been depicted in other PCI specifications. The header layout is optimized for performance on a serialized interconnect, driven by the requirement that the most time critical information be transferred first. For example, within the TLP header, the most significant byte of the address field is transferred first so that it may be used for early address decode.

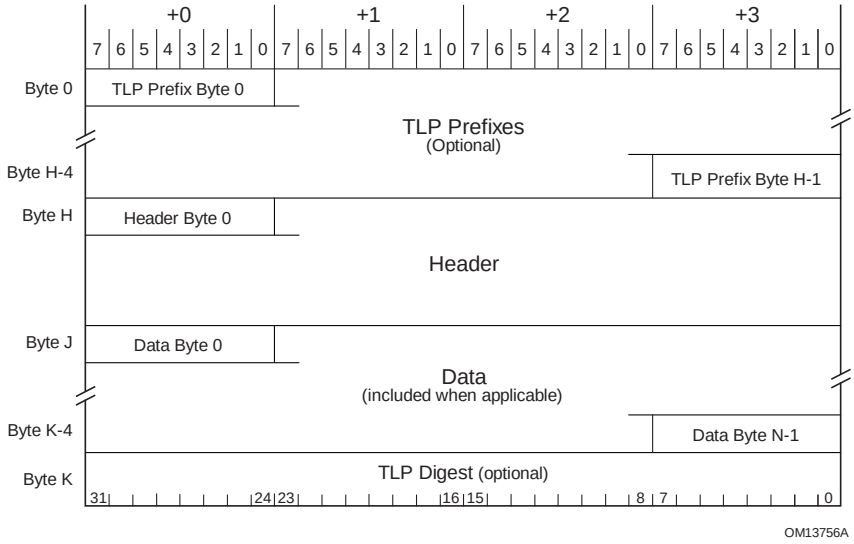


Figure 2-3: Generic TLP Format

Payload data within a TLP is depicted with the lowest addressed byte (byte J in Figure 2-3) shown to the upper left. Detailed layouts depicting data structure organization (such as the Configuration Space depictions in Chapter 7) retain the traditional PCI byte layout with the lowest addressed byte shown on the right. Regardless of depiction, all bytes are conceptually transmitted over the Link in increasing byte number order.

Depending on the type of a packet, the header for that packet will include some of the following types of fields:

- Format of the packet
- Type of the packet
- Length for any associated data
- Transaction Descriptor, including:
 - Transaction ID
 - Attributes
 - Traffic Class
- Address/routing information
- Byte Enables
- Message encoding
- Completion status

2.2. Transaction Layer Protocol - Packet Definition

PCI Express uses a packet based protocol to exchange information between the Transaction Layers of the two components communicating with each other over the Link. PCI Express supports the following basic transaction types: Memory, I/O, Configuration, and Messages. Two addressing formats for Memory Requests are supported: 32 bit and 64 bit.

Transactions are carried using Requests and Completions. Completions are used only where required, for example, to return read data, or to acknowledge Completion of I/O and Configuration Write Transactions. Completions are associated with their corresponding Requests by the value in the Transaction ID field of the Packet header.

All TLP fields marked Reserved (sometimes abbreviated as R) must be filled with all 0's when a TLP is formed. Values in such fields must be ignored by Receivers and forwarded unmodified by Switches. Note that for certain fields there are both specified and Reserved values – the handling of Reserved values in these cases is specified separately for each case.

2.2.1. Common Packet Header Fields

All Transaction Layer Packet (TLP) prefixes and headers contain the following fields (see Figure 2-4):

- Fmt[2:0] – Format of TLP (see Table 2-2) – bits 7:5 of byte 0
- Type[4:0] – Type of TLP – bits 4:0 of byte 0

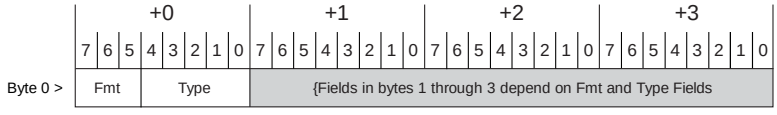


Figure 2-4: Fields Present in All TLPs

The Fmt field(s) indicate the presence of one or more TLP Prefixes and the Type field(s) indicates the associated TLP Prefix type(s).

The Fmt and Type fields of the TLP Header provide the information required to determine the size of the remaining part of the TLP Header, and if the packet contains a data payload following the header.

The Fmt, Type, TD, and Length fields of the TLP Header contain all information necessary to determine the overall size of the non-prefix portion of the TLP. The Type field, in addition to defining the type of the TLP also determines how the TLP is routed by a Switch. Different types of TLPs are discussed in more detail in the following sections.

- Permitted Fmt[2:0] and Type[4:0] field values are shown in Table 2-3.

All other encodings are Reserved (see Section 2.3).

- TC[2:0] – Traffic Class (see Section 2.2.6.6) – bits [6:4] of byte 1

- LN – 1b indicates that a Memory Request is an LN Read or LN Write, or that a Completion is an LN Completion.
- TH – 1b indicates the presence of TLP Processing Hints (TPH) in the TLP header and optional TPH TLP Prefix (if present) – bit 0 of byte 1 (see Section 2.2.7.1)
- Attr[1:0] – Attributes (see Section 2.2.6.3) – bits [5:4] of byte 2
- Attr[2] – Attribute (see Section 2.2.6.3) – bit 2 of byte 1
- TD – 1b indicates presence of TLP digest in the form of a single DW at the end of the TLP (see Section 2.2.3) – bit 7 of byte 2
- EP – indicates the TLP is poisoned (see Section 2.7) – bit 6 of byte 2
- Length[9:0] – Length of data payload in DW (see Table 2-4) – bits 1:0 of byte 2 concatenated with bits 7:0 of byte 3
 - TLP data must be 4-byte naturally aligned and in increments of 4-byte Double Words (DW).
 - Reserved for TLPs that do not contain or refer to data payloads, including Cpl, CplLk, and Messages (except as specified)

Comment [VS40]: ECN: Lightweight Notification (LN) Protocol

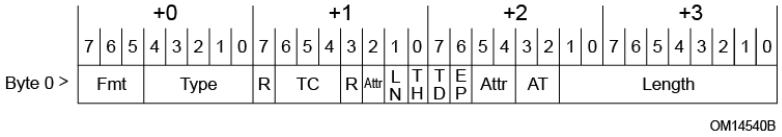


Figure 2-5: Fields Present in All TLP Headers

Comment [VS41]: Byte 1 bit 1 added per ECN: LN

Table 2-2: Fmt[2:0] Field Values

Fmt[2:0]	Corresponding TLP Format
000b	3 DW header, no data
001b	4 DW header, no data
010b	3 DW header, with data
011b	4 DW header, with data
100b	TLP Prefix
	All encodings not shown above are Reserved (see Section 2.3).

Table 2-3: Fmt[2:0] and Type[4:0] Field Encodings

TLP Type	Fmt [2:0][3] (b)	Type [4:0] (b)	Description
MRd	000 001	0 0000	Memory Read Request
MRdLk	000 001	0 0001	Memory Read Request-Locked
MWr	010 011	0 0000	Memory Write Request
IORd	000	0 0010	I/O Read Request
IOWr	010	0 0010	I/O Write Request
CfgRd0	000	0 0100	Configuration Read Type 0
CfgWr0	010	0 0100	Configuration Write Type 0
CfgRd1	000	0 0101	Configuration Read Type 1
CfgWr1	010	0 0101	Configuration Write Type 1
TCfgRd	000	1 1011	Deprecated TLP Type[4]
TCfgWr	010	1 1011	Deprecated TLP Type[4]
Msg	001	1 0$r_2r_1r_0$	Message Request – The sub-field r[2:0] specifies the Message routing mechanism (see Table 2-18).
MsgD	011	1 0$r_2r_1r_0$	Message Request with data payload – The sub-field r[2:0] specifies the Message routing mechanism (see Table 2-18).
Cpl	000	0 1010	Completion without Data – Used for I/O and Configuration Write Completions with any Completion Status. Also used for AtomicOp Completions and Read Completions (I/O, Configuration, or Memory) with Completion Status other than Successful Completion.
CplD	010	0 1010	Completion with Data – Used for Memory, I/O, and Configuration Read Completions. Also used for AtomicOp Completions.
CplLk	000	0 1011	Completion for Locked Memory Read without Data – Used only in error case.
CplDLk	010	0 1011	Completion for Locked Memory Read – otherwise like CplD.

[3] Requests with two Fmt[2:0] values shown can use either 32 bits (the first value) or 64 bits (the second value) Addressing Packet formats.

[4] Deprecated TLP Types: previously used for Trusted Configuration Space (TCS), which is no longer supported by this specification. If a Receiver does not implement TCS, the Receiver must treat such Requests as Malformed Packets.

TLP Type	Fmt [2:0][3] (b)	Type [4:0] (b)	Description
FetchAdd	010 011	0 1100	Fetch and Add AtomicOp Request
Swap	010 011	0 1101	Unconditional Swap AtomicOp Request
CAS	010 011	0 1110	Compare and Swap AtomicOp Request
LPrfx	100	$0L_3L_2L_1L_0$	Local TLP Prefix – The sub-field L[3:0] specifies the Local TLP Prefix type (see Table 2-35).
EPrfx	100	$1E_3E_2E_1E_0$	End-End TLP Prefix – The sub-field E[3:0] specifies the End-End TLP Prefix type (see Table 2-36).
			All encodings not shown above are Reserved (see Section 2.3).

Table 2-4: Length[9:0] Field Encoding

Length[9:0]	Corresponding TLP Data Payload Size
00 0000 0001b	1 DW
00 0000 0010b	2 DW
...	...
11 1111 1111b	1023 DW
00 0000 0000b	1024 DW

2.2.2. TLPs with Data Payloads - Rules

- Length is specified as an integral number of DW
- Length[9:0] is Reserved for all Messages except those which explicitly refer to a Data Length
 - Refer to the Message Code tables in Section 2.2.8.
- The Transmitter of a TLP with a data payload must not allow the data payload length as given by the TLP's Length field to exceed the length specified by the value in the Max_Payload_Size field of the Transmitter's Device Control register taken as an integral number of DW (see Section 7.8.4).
 - For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. The Max_Payload_Size settings in other Functions are ignored.
 - For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, a transmitted TLP's data payload must not exceed the common Max_Payload_Size setting.

- For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, a transmitted TLP's data payload must not exceed a Max_Payload_Size setting whose determination is implementation specific.
 - Transmitter implementations are encouraged to use the Max_Payload_Size setting from the Function that generated the transaction, or else the smallest Max_Payload_Size setting across all Functions.
 - Software should not set the Max_Payload_Size in different Functions to different values unless software is aware of the specific implementation.
- Note: Max_Payload_Size applies only to TLPs with data payloads; Memory Read Requests are not restricted in length by Max_Payload_Size. The size of the Memory Read Request is controlled by the Length field.

❑ The size of the data payload of a Received TLP as given by the TLP's Length field must not exceed the length specified by the value in the Max_Payload_Size field of the Receiver's Device Control register taken as an integral number of DW (see Section 7.8.4).

- Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, the TLP is a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).
- For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. The Max_Payload_Size settings in other Functions are ignored.
- For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the Receiver is required to check the TLP's data payload size against the common Max_Payload_Size setting.
- For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the Receiver is required to check the TLP's data payload against a Max_Payload_Size setting whose determination is implementation specific.
 - Receiver implementations are encouraged to use the Max_Payload_Size setting from the Function targeted by the transaction, or else the largest Max_Payload_Size setting across all Functions.
 - Software should not set the Max_Payload_Size in different Functions to different values unless software is aware of the specific implementation.

❑ For TLPs, that include data, the value in the Length field and the actual amount of data included in the TLP must match.

- Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, the TLP is a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

❑ The value in the Length field applies only to data – the TLP Digest is not included in the Length

- When a data payload is included in a TLP other than an AtomicOp Request or an AtomicOp Completion, the first byte of data following the header corresponds to the byte address closest to zero and the succeeding bytes are in increasing byte address sequence.
 - Example: For a 16-byte write to location 100h, the first byte following the header would be the byte to be written to location 100h, and the second byte would be written to location 101h, and so on, with the final byte written to location 10Fh.
- The data payload in AtomicOp Requests and AtomicOp Completions must be formatted such that the first byte of data following the TLP header is the least significant byte of the first data value, and subsequent bytes of data are strictly increasing in significance. With CAS Requests, the second data value immediately follows the first data value, and must be in the same format.
 - The endian format used by AtomicOp Completers to read and write data at the target location is implementation specific, and is permitted to be whatever the Completer determines is appropriate for the target memory (e.g., little endian, big endian, etc). Endian format capability reporting and controls for AtomicOp Completers are outside the scope of this specification.
 - Little endian example: For a 64-bit (8-byte) Swap Request targeting location 100h with the target memory in little endian format, the first byte following the header is written to location 100h, the second byte is written to location 101h, and so on, with the final byte written to location 107h. Note that before performing the writes, the Completer first reads the target memory locations so it can return the original value in the Completion. The byte address correspondence to the data in the Completion is identical to that in the Request.
 - Big endian example: For a 64-bit (8-byte) Swap Request targeting location 100h with the target memory in big endian format, the first byte following the header is written to location 107h, the second byte is written to location 106h, and so on, with the final byte written to location 100h. Note that before performing the writes, the Completer first reads the target memory locations so it can return the original value in the Completion. The byte address correspondence to the data in the Completion is identical to that in the Request.
 - Figure 2-6 shows little endian and big endian examples of Completer target memory access for a 64-bit (8-byte) FetchAdd. The bytes in the operands and results are numbered 0-7, with byte 0 being least significant and byte 7 being most significant. In each case, the Completer fetches the target memory operand using the appropriate endian format. Next, AtomicOp compute logic in the Completer performs the FetchAdd operation using the original target memory value and the "add" value from the FetchAdd Request. Finally, the Completer stores the FetchAdd result back to target memory using the same endian format used for the fetch.

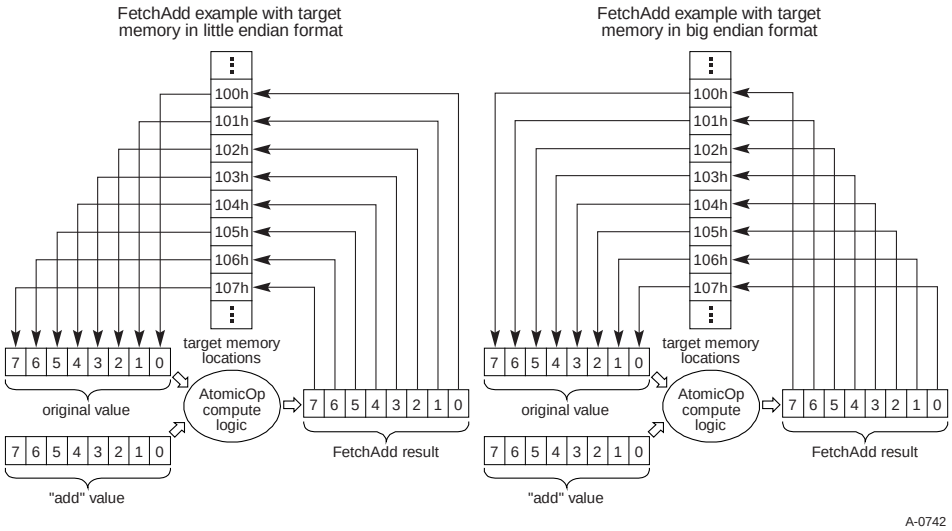


Figure 2-6: Examples of Completer Target Memory Access for FetchAdd

IMPLEMENTATION NOTE

Endian Format Support by RC AtomicOp Completers

One key reason for permitting an AtomicOp Completer to access target memory using an endian format of its choice is so that PCI Express devices targeting host memory with AtomicOps can interoperate with host software that uses atomic operation instructions (or instruction sequences). Some host environments have limited endian format support with atomic operations, and by supporting the "right" endian format(s), an RC AtomicOp Completer may significantly improve interoperability.

For an RC with AtomicOp Completer capability on a platform supporting little-endian-only processors, there is little envisioned benefit for the RC AtomicOp Completer to support any endian format other than little endian. For an RC with AtomicOp Completer capability on a platform supporting bi-endian processors, there may be benefit in supporting both big endian and little endian formats, and perhaps having the endian format configurable for different regions of host memory.

There is no PCI Express requirement that an RC AtomicOp Completer support the host processor's "native" format (if there is one), nor is there necessarily significant benefit to doing so. For example, some processors can use load-link/store-conditional or similar instruction sequences to do atomic operations in non-native endian formats and thus not need the RC AtomicOp Completer to support alternative endian formats.



IMPLEMENTATION NOTE

Maintaining Alignment in Data Payloads

Section 2.3.1.1 discusses rules for forming Read Completions respecting certain natural address boundaries. Memory Write performance can be significantly improved by respecting similar address boundaries in the formation of the Write Request. Specifically, forming Write Requests such that natural address boundaries of 64 or 128 bytes are respected will help to improve system performance.

2.2.3. TLP Digest Rules

- For any TLP, a value of 1b in the TD bit indicates the presence of the TLP Digest field including an ECRC value at the end of the TLP.
 - A TLP where the TD bit value does not correspond with the observed size (accounting for the data payload, if present) is a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).
- If an intermediate or ultimate PCI Express Receiver of the TLP does not support ECRC checking, the Receiver must ignore the TLP Digest[5].
 - If the Receiver of the TLP supports ECRC checking, the Receiver interprets the value in the TLP Digest field as an ECRC value, according to the rules in Section 2.7.1.

2.2.4. Routing and Addressing Rules

There are three principal mechanisms for TLP routing: address, ID, and implicit. This section defines the rules for the address and ID routing mechanisms. Implicit routing is used only with Message Requests, and is covered in Section 2.2.8.

[5] An exception is an Intermediate Receiver forwarding a Multicast TLP out an Egress Port with MC_Overlay enabled. See Section 6.14.5.

2.2.4.1. Address Based Routing Rules

- Address routing is used with Memory and I/O Requests.
- Two address formats are specified, a 64-bit format used with a 4 DW header (see Figure 2-7) and a 32-bit format used with a 3 DW header (see Figure 2-8).

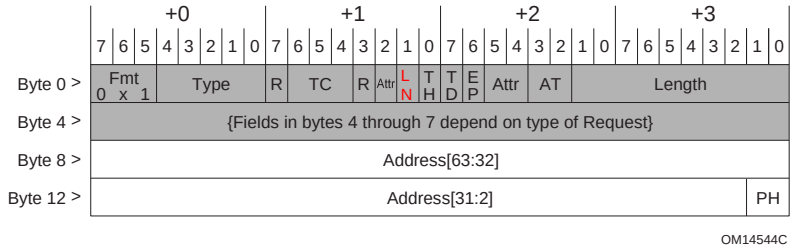


Figure 2-7: 64-bit Address Routing

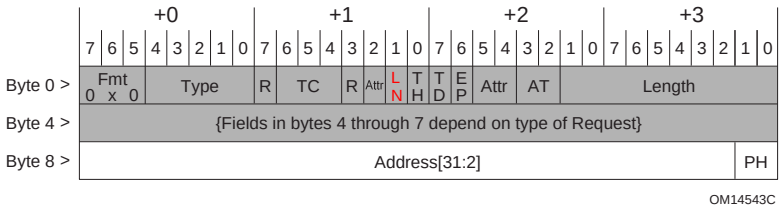


Figure 2-8: 32-bit Address Routing

- For Memory Read, Memory Write, and AtomicOp Requests, the Address Type (AT) field is encoded as shown in Table 2-5, with full descriptions contained in the *Address Translation Services Specification*. For all other Requests, the AT field is Reserved unless explicitly stated otherwise. LN Reads and LN Writes have special requirements. See Section 6.21.5.

Comment [VS42]: Errata A13

Table 2-5: Address Type (AT) Field Encodings

AT Coding (b)	Description
00	Default/Untranslated
01	Translation Request
10	Translated
11	Reserved

- Address mapping to the TLP header is shown in Table 2-6.

Table 2-6: Address Field Mapping

Address Bits	32-bit Addressing	64-bit Addressing
63:56	Not Applicable	Bits 7:0 of Byte 8
55:48	Not Applicable	Bits 7:0 of Byte 9
47:40	Not Applicable	Bits 7:0 of Byte 10
39:32	Not Applicable	Bits 7:0 of Byte 11
31:24	Bits 7:0 of Byte 8	Bits 7:0 of Byte 12
23:16	Bits 7:0 of Byte 9	Bits 7:0 of Byte 13
15:8	Bits 7:0 of Byte 10	Bits 7:0 of Byte 14
7:2	Bits 7:2 of Byte 11	Bits 7:2 of Byte 15

- Memory Read, Memory Write, and AtomicOp Requests can use either format.
 - For Addresses below 4 GB, Requesters must use the 32-bit format. The behavior of the receiver is not specified if a 64-bit format request addressing below 4 GB (i.e., with the upper 32 bits of address all 0) is received.
- I/O Read Requests and I/O Write Requests use the 32-bit format.
- All agents must decode all address bits in the header - address aliasing is not allowed.



IMPLEMENTATION NOTE

Prevention of Address Aliasing

For correct software operation, full address decoding is required even in systems where it may be known to the system hardware architect/designer that fewer than 64 bits of address are actually meaningful in the system.

2.2.4.2. ID Based Routing Rules

- ID routing is used with Configuration Requests, with ID Routed Messages, and with Completions. This specification defines Vendor_Defined Messages that are ID Routed (Section 2.2.8.6). Other specifications are permitted to define additional ID Routed Messages.[6]
- ID routing uses the Bus, Device, and Function Numbers (as applicable) to specify the destination for the TLP:
 - For non-ARI Routing IDs, Bus, Device, and (3-bit) Function Number to TLP header mapping is shown in Table 2-7.

[6] Currently, this includes the *Address Translation Services Specification*.

- For ARI Routing IDs, the Bus and (8-bit) Function Number to TLP header mapping is shown in Table 2-8.

- Two ID routing formats are specified, one used with a 4 DW header (see Figure 2-9) and one used with a 3 DW header (see Figure 2-10).
 - Header field locations are the same for both formats, and are given in Table 2-7 and Table 2-8.

Table 2-7: Header Field Locations for non-ARI ID Routing

Field	Header Location
Bus Number[7:0]	Bits 7:0 of Byte 8
Device Number[4:0]	Bits 7:3 of Byte 9
Function Number[2:0]	Bits 2:0 of Byte 9

Table 2-8: Header Field Locations for ARI ID Routing

Field	Header Location
Bus Number[7:0]	Bits 7:0 of Byte 8
Function Number[7:0]	Bits 7:0 of Byte 9

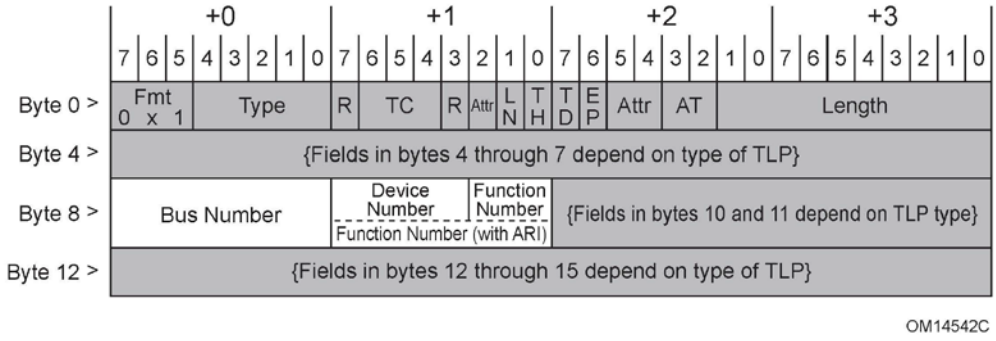


Figure 2-9: ID Routing with 4 DW Header

Comment [VS43]: LN bit added

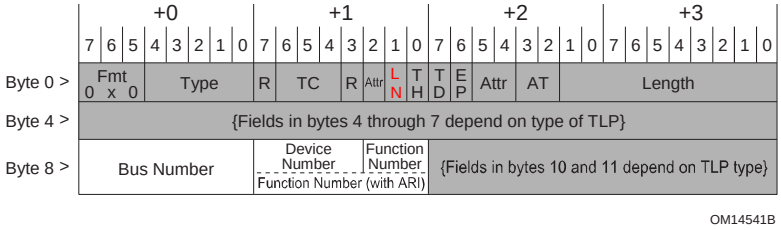


Figure 2-10: ID Routing with 3 DW Header

2.2.5. First/Last DW Byte Enables Rules

Byte Enables are included with Memory, I/O, and Configuration Requests. This section defines the corresponding rules. Byte Enables, when present in the Request header, are located in byte 7 of the header (see Figure 2-11). For Memory Read Requests that have the TH bit Set, the Byte Enable fields are repurposed to carry the ST[7:0] field (refer to Section 2.2.7.1 for details), and values for the Byte Enables are implied as defined below. The TH bit must only be set in Memory Read Requests when it is acceptable to complete those Requests as if all bytes for the requested data were enabled.

- For Memory Read Requests that have the TH bit Set, the following values are implied for the Byte Enables. See Section 2.2.7 for additional requirements.
 - If the Length field for this Request indicates a length of 1 DW, then the value for the 1st DW Byte Enables is implied to be 1111b and the value for the Last DW Byte Enables is implied to be 0000b.
 - If the Length field for this Request indicates a length of greater than 1 DW, then the value for the 1st DW Byte Enables and the Last DW Byte Enables is implied to be 1111b.



IMPLEMENTATION NOTE

Read Request with TPH to Non-Prefetchable Space

Memory Read Requests with the TH bit Set and that target Non-Prefetchable Memory Space should only be issued when it can be guaranteed that completion of such reads will not create undesirable side effects. See Section 7.5.2.1 for consideration of certain BARs that may have the Prefetchable bit Set even though they map some locations with read side-effects.

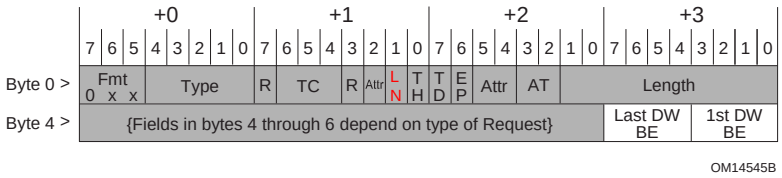


Figure 2-11: Location of Byte Enables in TLP Header

- The 1st DW BE[3:0] field contains Byte Enables for the first (or only) DW referenced by a Request.
 - If the Length field for a Request indicates a length of greater than 1 DW, this field must not equal 0000b.
- The Last DW BE[3:0] field contains Byte Enables for the last DW of a Request.
 - If the Length field for a Request indicates a length of 1 DW, this field must equal 0000b.
 - If the Length field for a Request indicates a length of greater than 1 DW, this field must not equal 0000b.
- For each bit of the Byte Enables fields:
 - a value of 0b indicates that the corresponding byte of data must not be written or, if non-prefetchable, must not be read at the Completer.
 - a value of 1b indicates that the corresponding byte of data must be written or read at the Completer.
- Non-contiguous Byte Enables (enabled bytes separated by non-enabled bytes) are permitted in the 1st DW BE field for all Requests with length of 1 DW.
 - Non-contiguous Byte Enable examples: 1010b, 0101b, 1001b, 1011b, 1101b
- Non-contiguous Byte Enables are permitted in both Byte Enables fields for QW aligned Memory Requests with length of 2 DW (1 QW).
- All non-QW aligned Memory Requests with length of 2 DW (1 QW) and Memory Requests with length of 3 DW or more must enable only bytes that are contiguous with the data between the first and last DW of the Request.
 - Contiguous Byte Enables examples:

 1st DW BE: 1100b, Last DW BE: 0011b

 1st DW BE: 1000b, Last DW BE: 0111b
- Table 2-9 shows the correspondence between the bits of the Byte Enables fields, their location in the Request header, and the corresponding bytes of the referenced data.

Table 2-9: Byte Enables Location and Correspondence

Byte Enables	Header Location	Affected Data Byte[7]
1st DW BE[0]	Bit 0 of Byte 7	Byte 0
1st DW BE[1]	Bit 1 of Byte 7	Byte 1
1st DW BE[2]	Bit 2 of Byte 7	Byte 2
1st DW BE[3]	Bit 3 of Byte 7	Byte 3
Last DW BE[0]	Bit 4 of Byte 7	Byte N-4
Last DW BE[1]	Bit 5 of Byte 7	Byte N-3
Last DW BE[2]	Bit 6 of Byte 7	Byte N-2
Last DW BE[3]	Bit 7 of Byte 7	Byte N-1

- A Write Request with a length of 1 DW with no bytes enabled is permitted, and has no effect at the Completer unless otherwise specified.



IMPLEMENTATION NOTE

Zero-Length Write

A Memory Write Request of 1 DW with no bytes enabled, or "zero-length Write," may be used by devices under certain protocols, in order to achieve an intended side effect. One example is LN protocol. See Section 6.21.

Comment [VS44]: ECN: Lightweight Notification (LN) Protocol

- If a Read Request of 1 DW specifies that no bytes are enabled to be read (1st DW BE[3:0] field = 0000b), the corresponding Completion must specify a Length of 1 DW, and include a data payload of 1 DW

The contents of the data payload within the Completion packet is unspecified and may be any value

- Receiver/Completer behavior is undefined for a TLP violating the Byte Enables rules specified in this section.
- Receivers may optionally check for violations of the Byte Enables rules specified in this section. If a Receiver implementing such checks determines that a TLP violates one or more Byte Enables rules, the TLP is a Malformed TLP. These checks are independently optional (see Section 6.2.3.4).
 - If Byte Enables rules are checked, a violation is a reported error associated with the Receiving Port (see Section 6.2).

[7] Assuming the data referenced is N bytes in length (Byte 0 to Byte N-1). Note that last DW Byte Enables are used only if the data length is greater than one DW.

IMPLEMENTATION NOTE

Zero-Length Read

A Memory Read Request of 1 DW with no bytes enabled, or "zero-length Read," may be used by devices as a type of flush Request. For a Requester, the flush semantic allows a device to ensure that previously issued Posted Writes have been completed at their PCI Express destination. To be effective in all cases, the address for the zero-length Read must target the same device as the Posted Writes that are being flushed. One recommended approach is using the same address as one of the Posted Writes being flushed.

The flush semantic has wide application, and all Completers must implement the functionality associated with this semantic. Since a Requester may use the flush semantic without comprehending the characteristics of the Completer, Completers must ensure that zero-length reads do not have side-effects. This is really just a specific case of the rule that in a non-prefetchable space, non-enabled bytes must not be read at the Completer. Note that the flush applies only to traffic in the same Traffic Class as the zero-length Read.

2.2.6. Transaction Descriptor

2.2.6.1. Overview

The Transaction Descriptor is a mechanism for carrying Transaction information between the Requester and the Completer. Transaction Descriptors are composed of three fields:

- Transaction ID – identifies outstanding Transactions
- Attributes field – specifies characteristics of the Transaction
- Traffic Class (TC) field – associates Transaction with type of required service

Figure 2-12 shows the fields of the Transaction Descriptor. Note that these fields are shown together to highlight their relationship as parts of a single logical entity. The fields are not contiguous in the packet header.

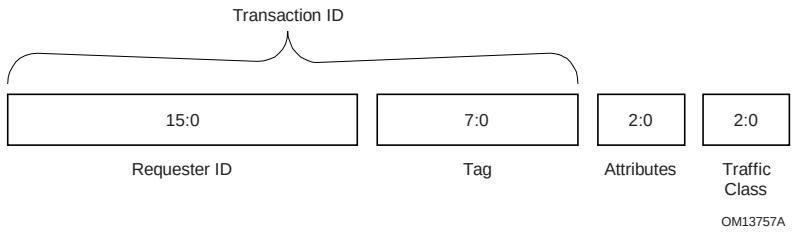


Figure 2-12: Transaction Descriptor

2.2.6.2. Transaction Descriptor – Transaction ID Field

The Transaction ID field consists of two major sub-fields: Requester ID and Tag as shown in Figure 2-13.

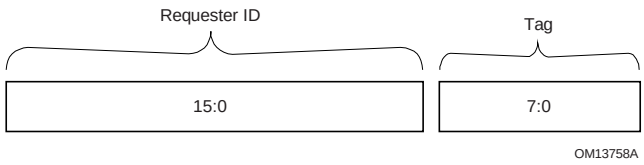


Figure 2-13: Transaction ID

- Tag[7:0] is an 8-bit field generated by each Requester, and it must be unique for all outstanding Requests that require a Completion for that Requester
 - Receivers/Completers must handle correctly 8-bit Tag values regardless of the setting of their Extended Tag Field Enable bit. Refer to the *PCI Express to PCI/PCI-X Bridge Specification* for details on the bridge handling of extended tags
 - If the Extended Tag Field Enable bit (see Section 7.8.4) is clear, the maximum number of outstanding Requests per Function shall be limited to 32, and only the lower 5 bits of the Tag field are used with the remaining upper 3 bits required to be 000b
 - If the Extended Tag Field Enable bit (see Section 7.8.4) is set, the maximum is increased to 256, and the entire Tag field is used
 - The initial value of the Extended Tag Field Enable bit (see Section 7.8.4) is implementation specific
 - Receiver/Completer behavior is undefined if multiple uncompleted Requests are issued non-unique Tag values
 - If Phantom Function Numbers are used to extend the number of outstanding requests, the combination of the Phantom Function Number and the Tag field must be unique for all outstanding Requests that require a Completion for that Requester.
- For Posted Requests with the TH bit Set, the Tag[7:0] field is repurposed for the ST[7:0] field (refer to Section 2.2.7.1 for details). For Posted Requests with the TH bit Clear, the Tag[7:0] field is undefined and may contain any value. (Refer to Section 2.2.8.6 for exceptions to this rule for certain Vendor_Defined Messages.)
 - For Posted Requests with the TH field Clear, the value in the Tag[7:0] field must not affect Receiver processing of the Request
 - For Posted Requests with the TH bit Set, the value in the ST[7:0] field may affect Completer processing of the Request (refer to 2.2.7.1 for details).
- Requester ID and Tag combined form a global identifier, i.e., Transaction ID for each Transaction within a Hierarchy.
- Transaction ID is included with all Requests and Completions.
- The Requester ID is a 16-bit value that is unique for every PCI Express Function within a Hierarchy.

- Functions must capture the Bus and Device Numbers[8] supplied with all Type 0 Configuration Write Requests completed by the Function and supply these numbers in the Bus and Device Number fields of the Requester ID[9] for all Requests initiated by the Device/Function.

 Exception: The assignment of Bus and Device Numbers to the Devices within a Root Complex, and Device Numbers to the Downstream Ports within a Switch, may be done in an implementation specific way.

 Note that the Bus Number and Device Number[10] may be changed at run time, and so it is necessary to re-capture this information with each and every Configuration Write Request.

- When generating Requests on their own behalf (for example, for error reporting), Switches must use the Requester ID associated with the primary side of the bridge logically associated with the Port (see Section 7.1) causing the Request generation.

- Prior to the initial Configuration Write to a Function, the Function is not permitted to initiate Non-Posted Requests (A valid Requester ID is required to properly route the resulting completions).

 - Exception: Functions within a Root Complex are permitted to initiate Requests prior to software-initiated configuration for accesses to system boot device(s).

 Note that this rule and the exception are consistent with the existing PCI model for system initialization and configuration.

- Each Function associated with a Device must be designed to respond to a unique Function Number for Configuration Requests addressing that Device.
 Note: Each non-ARI Device may contain up to eight Functions. Each ARI Device may contain up to 256 Functions.

- A Switch must forward Requests without modifying the Transaction ID

- In some circumstances, a PCI Express to PCI/PCI-X Bridge is required to generate Transaction IDs for requests it forwards from a PCI or PCI-X bus.



IMPLEMENTATION NOTE

Increasing the Number of Outstanding Requests

To increase the maximum possible number of outstanding Requests requiring Completion beyond 256, a device may, if the Phantom Functions Enable bit is set (see Section 7.8.4), use Function Numbers not assigned to implemented Functions to logically extend the Tag identifier. For a single-Function device, this can allow up to an 8-fold increase in the maximum number of outstanding Requests.

Unclaimed Function Numbers are referred to as Phantom Function Numbers (PFN).

[8] In ARI Devices, Functions are only required to capture the Bus Number. ARI Devices are permitted to retain the captured Bus Number on either a per-Device or a per-Function basis. If the captured Bus Number is retained on a per-Device basis, all Functions are required to update and use the common Bus Number.

[9] An ARI Requester ID does not contain a Device Number field. See Section 2.2.4.2.

[10] With ARI Devices, only the Bus Number can change.

2.2.6.3. Transaction Descriptor – Attributes Field

The Attributes field is used to provide additional information that allows modification of the default handling of Transactions. These modifications apply to different aspects of handling the Transactions within the system, such as:

- Ordering
- Hardware coherency management (snoop)

Note that attributes are hints that allow for optimizations in the handling of traffic. Level of support is dependent on target applications of particular PCI Express peripherals and platform building blocks. Refer to PCI-X 2.0 for additional details regarding these attributes. Note that attribute bit 2 is not adjacent to bits 1 and 0 (see Figure 2-15 and Figure 2-16).

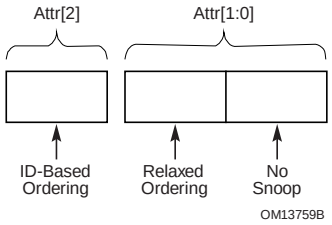


Figure 2-14: Attributes Field of Transaction Descriptor

2.2.6.4. Relaxed Ordering and ID-Based Ordering Attributes

Table 2-10 defines the states of the Relaxed Ordering and ID-Based Ordering attribute fields. These attributes are discussed in Section 2.4. Note that Relaxed Ordering and ID-Based Ordering attributes are not adjacent in location (see Figure 2-5).

Table 2-10: Ordering Attributes

Attribute Bit [2]	Attribute Bit [1]	Ordering Type	Ordering Model
0	0	Default Ordering	PCI Strongly Ordered Model
0	1	Relaxed Ordering	PCI-X Relaxed Ordering Model
1	0	ID-Based Ordering	Independent ordering based on Requester/Completer ID
1	1	Relaxed Ordering plus ID-Based Ordering	Logical "OR" of Relaxed Ordering and IDO

Attribute bit [1] is not applicable and must be set to 0b for Configuration Requests, I/O Requests, Memory Requests that are Message Signaled Interrupts, and Message Requests (except where specifically permitted).

Attribute bit [2], IDO, is Reserved for Configuration Requests and I/O Requests. IDO is not Reserved for all Memory Requests, including Message Signaled Interrupts. IDO is not Reserved for Message Requests unless specifically prohibited. A Requester is permitted to set IDO only if the IDO Request Enable bit in the Device Control 2 register is set.

The value of the IDO bit must not be considered by Receivers when determining if a TLP is a Malformed Packet.

A Completer is permitted to set IDO only if the IDO Completion Enable bit in the Device Control 2 register is set. It is not required to copy the value of IDO from the Request into the Completion(s) for that Request. If the Completer has IDO enabled, it is recommended that the Completer set IDO for all Completions, unless there is a specific reason not to (see Appendix E)

A Root Complex that supports forwarding TLPs peer-to-peer between Root Ports is not required to preserve the IDO bit from the Ingress to Egress Port.

2.2.6.5. No Snoop Attribute

Table 2-11 defines the states of the No Snoop attribute field. Note that the No Snoop attribute does not alter Transaction ordering.

Table 2-11: Cache Coherency Management Attribute

No Snoop Attribute (b)	Cache Coherency Management Type	Coherency Model
0	Default	Hardware enforced cache coherency expected
1	No Snoop	Hardware enforced cache coherency not expected

This attribute is not applicable and must be set to 0b for Configuration Requests, I/O Requests, Memory Requests that are Message Signaled Interrupts, and Message Requests (except where specifically permitted).

2.2.6.6. Transaction Descriptor – Traffic Class Field

The Traffic Class (TC) is a 3-bit field that allows differentiation of transactions into eight traffic classes.

Together with the PCI Express Virtual Channel support, the TC mechanism is a fundamental element for enabling differentiated traffic servicing. Every PCI Express Transaction Layer Packet uses TC information as an invariant label that is carried end to end within the PCI Express fabric. As the packet traverses across the fabric, this information is used at every Link and within each Switch element to make decisions with regards to proper servicing of the traffic. A key aspect of servicing is the routing of the packets based on their TC labels through corresponding Virtual Channels. Section 2.5 covers the details of the VC mechanism.

Table 2-12 defines the TC encodings.

Table 2-12: Definition of TC Field Encodings

TC Field Value (b)	Definition
000	TC0: Best Effort service class (General Purpose I/O) (Default TC – must be supported by every PCI Express device)
001 – 111	TC1-TC7: Differentiated service classes (Differentiation based on Weighted-Round-Robin and/or Priority)

It is up to the system software to determine TC labeling and TC/VC mapping in order to provide differentiated services that meet target platform requirements.

The concept of Traffic Class applies only within the PCI Express interconnect fabric. Specific requirements of how PCI Express TC service policies are translated into policies on non-PCI Express interconnects is outside of the scope of this specification.

2.2.7. Memory, I/O, and Configuration Request Rules

The following rule applies to all Memory, I/O, and Configuration Requests. Additional rules specific to each type of Request follow.

- All Memory, I/O, and Configuration Requests include the following fields in addition to the common header fields:
 - Requester ID[15:0] and Tag[7:0], forming the Transaction ID
 - Last DW BE[3:0] and 1st DW BE[3:0]. For Memory Read Requests and AtomicOp Requests with the TH bit Set, the byte location for the Last DW BE[3:0] and 1st DW BE [3:0] fields in the header are repurposed to carry ST[7:0] field. For Memory Read Requests with the TH bit Clear, see Section 2.2.5 for First/Last DW Byte Enable Rules. For AtomicOp Requests with TH bit Set, the values for the DW BE fields are implied to be Reserved. For AtomicOp Requests with TH bit clear, the DW BE fields are Reserved.

For Memory Requests, the following rules apply:

- Memory Requests route by address, using either 64-bit or 32-bit Addressing (see Figure 2-15 and Figure 2-16)
- For Memory Read Requests, Length must not exceed the value specified by Max_Read_Request_Size (see Section 7.8.4)
- For AtomicOp Requests, architected operand sizes and their associated Length field values are specified in Table 2-13. If a Completer supports AtomicOps, the following rules apply. The Completer must check the Length field value. If the value does not match an architected value, the Completer must handle the TLP as a Malformed TLP. Otherwise, if the value does not match an operand size that the Completer supports, the Completer must handle the TLP as an Unsupported Request (UR). This is a reported error associated with the Receiving Port (see Section 6.2).

Comment [VS45]: Errata A7

Table 2-13: Length Field Values for AtomicOp Requests

AtomicOp Request	Length Field Value for Architected Operand Sizes		
	32 Bits	64 Bits	128 Bits
FetchAdd, Swap	1 DW	2 DW	N/A
CAS	2 DW	4 DW	8 DW

 - A FetchAdd Request contains one operand, the "add" value.
 - A Swap Request contains one operand, the "swap" value.
 - A CAS Request contains two operands. The first in the data area is the "compare" value, and the second is the "swap" value.
- For AtomicOp Requests, the Address must be naturally aligned with the operand size. The Completer must check for violations of this rule. If a TLP violates this rule, the TLP is a Malformed TLP. This is a reported error associated with the Receiving Port (see Section 6.2).

- Requests must not specify an Address/Length combination which causes a Memory Space access to cross a 4-KB boundary.
 - Receivers may optionally check for violations of this rule. If a Receiver implementing this check determines that a TLP violates this rule, the TLP is a Malformed TLP.
 - If checked, this is a reported error associated with the Receiving Port (see Section 6.2).
 - For AtomicOp Requests, the mandatory Completer check for natural alignment of the Address (see above) already guarantees that the access will not cross a 4-KB boundary, so a separate 4-KB boundary check is not necessary.
 - Note that if a 4-KB boundary check is performed for AtomicOp CAS Requests, this check must comprehend that the TLP Length value is based on the size of two operands, whereas the access to Memory Space is based on the size of one operand.

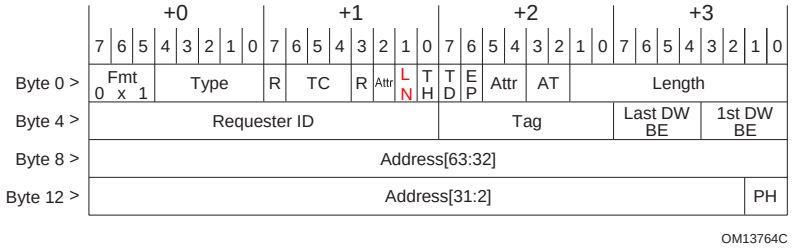


Figure 2-15: Request Header Format for 64-bit Addressing of Memory

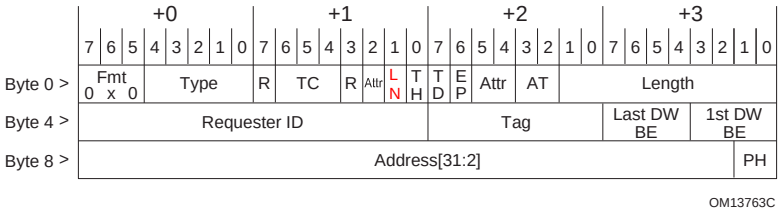


Figure 2-16: Request Header Format for 32-bit Addressing of Memory

IMPLEMENTATION NOTE

Generation of 64-bit Addresses

It is strongly recommended that PCI Express Endpoints be capable of generating the full range of 64-bit addresses. However, if a PCI Express Endpoint supports a smaller address range, and is unable to reach the full address range required by a given platform environment, the corresponding Device Driver must ensure that all Memory Transaction target buffers fall within the address range supported by the Endpoint. The exact means of ensuring this is platform and operating system specific, and beyond the scope of this specification.

For I/O Requests, the following rules apply:

- I/O Requests route by address, using 32-bit Addressing (see Figure 2-17)
- I/O Requests have the following restrictions:
 - TC[2:0] must be 000b
 - LN is not applicable to I/O Requests and the bit is Reserved
 - TH is not applicable to I/O Request and the bit is Reserved
 - Attr[2] is Reserved
 - Attr[1:0] must be 00b
 - AT[1:0] must be 00b. Receivers are not required or encouraged to check this.
 - Length[9:0] must be 00 0000 0001b
 - Last DW BE[3:0] must be 0000b

Receivers may optionally check for violations of these rules (but must not check reserved bits). These checks are independently optional (see Section 6.2.3.4). If a Receiver implementing these checks determines that a TLP violates these rules, the TLP is a Malformed TLP.

- If checked, this is a reported error associated with the Receiving Port (see Section 6.2).

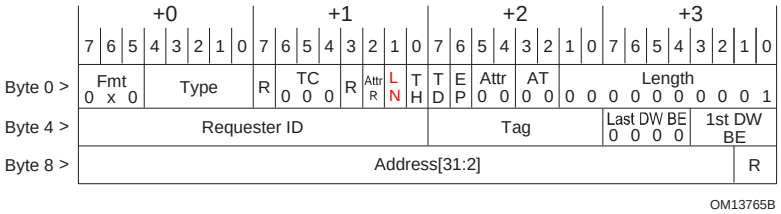


Figure 2-17: Request Header Format for I/O Transactions

Comment [VS46]: ECN: Lightweight Notification (LN) Protocol

Comment [VS47]: Errata A13

For Configuration Requests, the following rules apply:

- Configuration Requests route by ID, and use a 3 DW header
- In addition to the header fields included in all Memory, I/O, and Configuration Requests and the ID routing fields, Configuration Requests contain the following additional fields (see Figure 2-18)
 - Register Number[5:0]
 - Extended Register Number[3:0]
- Configuration Requests have the following restrictions:
 - TC[2:0] must be 000b
 - LN is not applicable to Configuration Requests and the bit is Reserved
 - TH is not applicable to Configuration Requests and the bit is Reserved
 - Attr[2] is Reserved
 - Attr[1:0] must be 00b
 - AT[1:0] must be 00b. Receivers are not required or encouraged to check this.
 - Length[9:0] must be 00 0000 0001b
 - Last DW BE[3:0] must be 0000b

Receivers may optionally check for violations of these rules (but must not check reserved bits). These checks are independently optional (see Section 6.2.3.4). If a Receiver implementing these checks determines that a TLP violates these rules, the TLP is a Malformed TLP.

- If checked, this is a reported error associated with the Receiving Port (see Section 6.2).

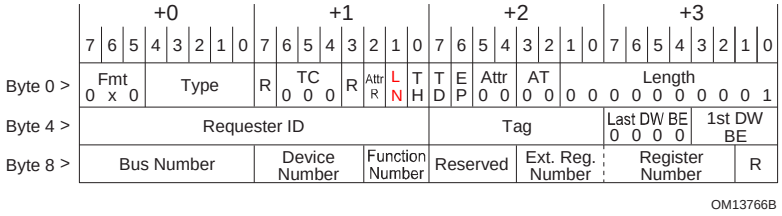


Figure 2-18: Request Header Format for Configuration Transactions

Message Signaled Interrupt (MSI/MSI-X) mechanisms use Memory Write Requests to represent interrupt Messages (see Section 6.1.4). The Request format used for MSI/MSI-X transactions is identical to the Memory Write Request format defined above, and MSI/MSI-X Requests are indistinguishable from memory writes with regard to ordering, Flow Control, and data integrity.

Comment [VS48]: ECN: Lightweight Notification (LN) Protocol

Comment [VS49]: Errata A13

2.2.7.1. TPH Rules

- Two formats are specified for TPH. The Baseline TPH format (see Figure 2-20 and Figure 2-21) must be used for all Requests that provide TPH. The format with the Optional TPH TLP Prefix extends the TPH fields (see Figure 2-19) to provide additional bits for the Steering Tag (ST) field.

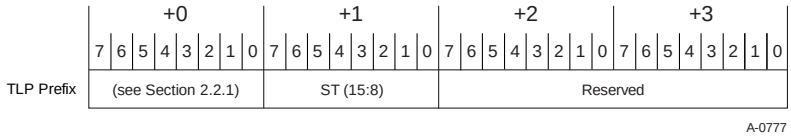


Figure 2-19: TPH TLP Prefix

- The Optional TPH TLP Prefix is used to extend the TPH fields.
 - The presence of a TPH TLP Prefix is determined by decoding byte 0.

Table 2-14: TPH TLP Prefix Bit Mapping

Fields	TPH TLP Prefix
ST(15:8)	Bits 7:0 of byte 1
Reserved	Bits 7:0 of byte 2
Reserved	Bits 7:0 of byte 3

- For Requests that target Memory Space, a value of 1b in the TH bit indicates the presence of TPH in the TLP header and optional TPH TLP Prefix (if present).
 - The TH bit must be set for Requests that provide TPH
 - The TH bit must be set for Requests with a TPH TLP Prefix
 - The TH bit is not applicable and is Reserved for all other Requests.
- The Processing Hints (PH) fields mapping is shown in Figure 2-20, Figure 2-21, and Table 2-15.

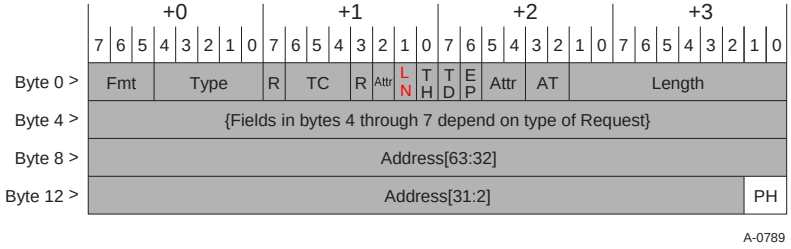


Figure 2-20: Location of PH[1:0] in a 4 DW Request Header

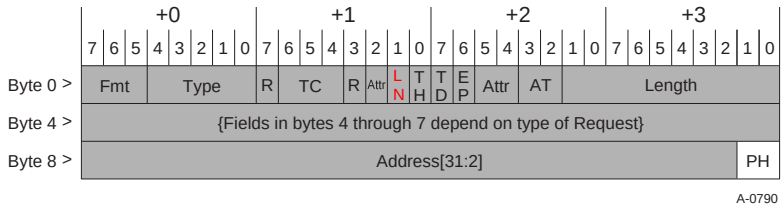


Figure 2-21: Location of PH[1:0] in a 3 DW Request Header

Table 2-15: Location of PH[1:0] in TLP Header

PH	32-bit Addressing	64-bit Addressing
1:0	Bits 1:0 of Byte 11	Bits 1:0 of Byte 15

- The PH[1:0] field provides information about the data access patterns and is defined as described in Table 2-16.

Table 2-16: Processing Hint Encoding

PH[1:0] (b)	Processing Hint	Description
00	Bi-directional data structure	Indicates frequent read and/or write access to data by Host and device
01	Requester	Indicates frequent read and/or write access to data by device
10	Target	Indicates frequent read and/or write access to data by Host
11	Target with Priority	Indicates frequent read and/or write access by Host and indicates high temporal locality for accessed data

- The Steering Tag (ST) fields are mapped to the TLP header as shown in Figure 2-22, Figure 2-23, and Table 2-17.

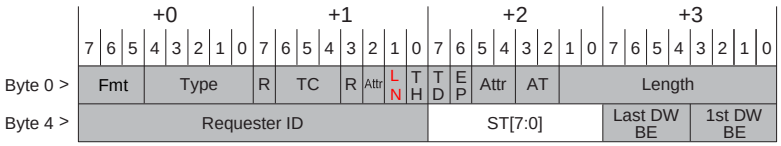


Figure 2-22: Location of ST[7:0] in the Memory Write Request Header

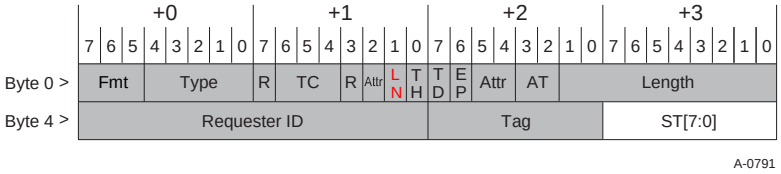


Figure 2-23: Location of ST[7:0] in Memory Read and AtomicOp Request Headers

Table 2-17: Location of ST[7:0] in TLP Headers

ST Bits	Memory Write Request	Memory Read Request or AtomicOp Request
7:0	Bits 7:0 of Byte 6	Bits 7:0 of Byte 7

- ST[7:0] field carries the Steering Tag value

A value of all zeroes indicates no Steering Tag preference

A total of 255 unique Steering Tag values are provided

- A Function that does not support the TPH Completer or Routing capability and receives a transaction with the TH bit Set is required to ignore the TH bit and handle the Request in the same way as Requests of the same transaction type without the TH bit Set.

2.2.8. Message Request Rules

This document defines the following groups of Messages:

- INTx Interrupt Signaling
- Power Management
- Error Signaling
- Locked Transaction Support
- Slot Power Limit Support
- Vendor-Defined Messages
- LTR Messages
- OBFF Messages
- DRS Messages
- FRS Messages
- PTM Messages

The following rules apply to all Message Requests. Additional rules specific to each type of Message follow.

- All Message Requests include the following fields in addition to the common header fields (see Figure 2-24):
 - Requester ID[15:0] and Tag[7:0], forming the Transaction ID.
 - Message Code[7:0] – Specifies the particular Message embodied in the Request.
- All Message Requests use the Msg or MsgD Type field encoding~~, except for the Vendor_Defined Messages, which can use either Msg or MsgD, and the Set_Slot_Power_Limit Message, which uses MsgD~~.
- The Message Code field must be fully decoded (Message aliasing is not permitted).
- The Attr[2] field is not Reserved unless specifically indicated as Reserved.
- Except as noted, the Attr[1:0] field is Reserved.
- LN is not applicable to Message Requests and the bit is Reserved.
- Except as noted, TH is not applicable to Message Requests and the bit is Reserved.
- AT[1:0] must be 00b. Receivers are not required or encouraged to check this.
- Except as noted, bytes 8 through 15 are Reserved.
- Message Requests are posted and do not require Completion.
- Message Requests follow the same ordering rules as Memory Write Requests.

Comment [VS50]: RC_ECN_RN_27_Aug_2013

Comment [VS51]: RC_ECN_RN_27_Aug_2013

Comment [VS52]: ECN: Precision Time Measurement

Comment [VS53]: Errata B4 applied after ECN: Precision Time Measurement (PTM)

Comment [VS54]: ECN: Lightweight Notification (LN) Protocol

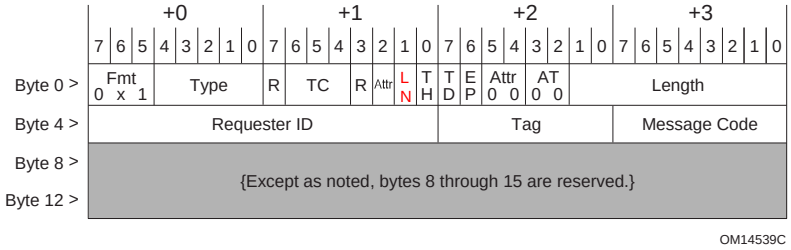


Figure 2-24: Message Request Header

In addition to address and ID routing, Messages support several other routing mechanisms. These mechanisms are referred to as "implicit" because no address or ID specifies the destination, but rather the destination is implied by the routing type. The following rules cover Message routing mechanisms:

- Message routing is determined using the r[2:0] sub-field of the Type field
 - Message Routing r[2:0] values are defined in Table 2-18
 - Permitted values are defined in the following sections for each Message

Table 2-18: Message Routing

r[2:0] (b)	Description	Bytes 8 Through 15[11]
000	Routed to Root Complex	Reserved
001	Routed by Address[12]	Address
010	Routed by ID	See Section 2.2.4.2
011	Broadcast from Root Complex	Reserved
100	Local - Terminate at Receiver	Reserved
101	Gathered and routed to Root Complex[13]	Reserved
110-111	Reserved - Terminate at Receiver	Reserved

[11] Except as noted, e.g., Vendor_Defined Messages.

[12] Note that no Messages defined in this document use Address routing.

[13] This routing type is used only for PME_TO_Ack, and is described in Section 5.3.3.2.1.

2.2.8.1. INTx Interrupt Signaling - Rules

A Message Signaled Interrupt (MSI or MSI-X) is the preferred interrupt signaling mechanism in PCI Express (see Section 6.1). However, in some systems, there may be Functions that cannot support the MSI or MSI-X mechanisms. The INTx virtual wire interrupt signaling mechanism is used to support Legacy Endpoints and PCI Express/PCI(-X) Bridges in cases where the MSI or MSI-X mechanisms cannot be used. Switches must support this mechanism. The following rules apply to the INTx interrupt signaling mechanism:

- The INTx mechanism uses eight distinct Messages (see Table 2-19).
- Assert_INTx/Deassert_INTx Messages do not include a data payload (TLP Type is Msg).
- The Length field is Reserved.
- With Assert_INTx/Deassert_INTx Messages, the Function Number field in the Requester ID must be 0. Note that the Function Number field is a different size for non-ARI and ARI Requester IDs.
- Assert_INTx/Deassert_INTx Messages are only issued by Upstream Ports.
 - Receivers may optionally check for violations of this rule. If a Receiver implementing this check determines that an Assert_INTx/Deassert_INTx violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).
- Assert_INTx and Deassert_INTx interrupt Messages must use the default Traffic Class designator (TC0). Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

Table 2-19: INTx Mechanism Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support[14]				Description/Comments
			RC	Ep	Sw	Br	
Assert_INTA	0010 0000	100	All:				Assert INTA virtual wire Note: These Messages are used for PCI 3.0 compatible INTx emulation.
			r		tr		
			As Required:				
				t		t	
Assert_INTB	0010 0001	100	All:				Assert INTB virtual wire
			r		tr		
			As Required:				
				t		t	
Assert_INTC	0010 0010	100	All:				Assert INTC virtual wire
			r		tr		
			As Required:				
				t		t	
Assert_INTD	0010 0011	100	All:				Assert INTD virtual wire
			r		tr		
			As Required:				
				t		t	
Deassert_INTA	0010 0100	100	All:				De-assert INTA virtual wire
			r		tr		
			As Required:				
				t		t	

[14] Abbreviations:
RC = Root Complex
Sw = Switch (only used with “Link” routing)
Ep = Endpoint
Br = PCI Express (primary) to PCI/PCI-X (secondary) Bridge
r = Supports as Receiver
t = Supports as Transmitter

Note that Switches must support passing Messages on all legal routing paths. Only Messages specifying Local (100b) routing or a Reserved routing value are terminated locally at the Receiving Port on a Switch.

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support[13]				Description/Comments
			RC	Ep	Sw	Br	
Deassert_INTB	0010 0101	100	All:				De-assert INTB virtual wire
			r		tr		
			As Required:				
				t		t	
Deassert_INTC	0010 0110	100	All:				De-assert INTC virtual wire
			r		tr		
			As Required:				
				t		t	
Deassert_INTD	0010 0111	100	All:				De-assert INTD virtual wire
			r		tr		
			As Required:				
				t		t	

The Assert_INTx/Deassert_INTx Message pairs constitute four "virtual wires" for each of the legacy PCI interrupts designated A, B, C, and D. The following rules describe the operation of these virtual wires:

- The components at both ends of each Link must track the logical state of the four virtual wires using the Assert/Deassert Messages to represent the active and inactive transitions (respectively) of each corresponding virtual wire.
 - An Assert_INTx represents the active going transition of the INTx (x = A, B, C, or D) virtual wire
 - A Deassert_INTx represents the inactive going transition of the INTx (x = A, B, C, or D) virtual wire
- When the local logical state of an INTx virtual wire changes at an Upstream Port, the Port must communicate this change in state to the Downstream Port on the other side of the same Link using the appropriate Assert_INTx or Deassert_INTx Message.

Note: Duplicate Assert_INTx/Deassert_INTx Messages have no effect, but are not errors.

- INTx Interrupt Signaling is disabled when the Interrupt Disable bit of the Command register (see Section 7.5.1.1) is set to 1b.
 - Any INTx virtual wires that are active when the Interrupt Disable bit is set must be deasserted by transmitting the appropriate Deassert_INTx Message(s).
- Virtual and actual PCI to PCI Bridges must map the virtual wires tracked on the secondary side of the Bridge according to the Device Number of the device on the secondary side of the Bridge, as shown in Table 2-20.
- Switches must track the state of the four virtual wires independently for each Downstream Port, and present a "collapsed" set of virtual wires on its Upstream Port.

- If a Switch Downstream Port goes to DL_Down status, the INTx virtual wires associated with that Port must be deasserted, and the Switch Upstream Port virtual wire state updated accordingly.
 - If this results in de-assertion of any Upstream INTx virtual wires, the appropriate Deassert_INTx Message(s) must be sent by the Upstream Port.
- The Root Complex must track the state of the four INTx virtual wires independently for each of its Downstream Ports, and map these virtual signals to system interrupt resources.
 - Details of this mapping are system implementation specific.
- If a Downstream Port of the Root Complex goes to DL_Down status, the INTx virtual wires associated with that Port must be deasserted, and any associated system interrupt resource request(s) must be discarded.

Comment [VS55]: Errata B19

Table 2-20: Bridge Mapping for INTx Virtual Wires

Device Number for Device on Secondary Side of Bridge (Interrupt Source)	INTx Virtual Wire on Secondary Side of Bridge	Mapping to INTx Virtual Wire on Primary Side of Bridge
0,4,8,12,16,20,24,28	INTA	INTA
	INTB	INTB
	INTC	INTC
	INTD	INTD
1,5,9,13,17,21,25,29	INTA	INTB
	INTB	INTC
	INTC	INTD
	INTD	INTA
2,6,10,14,18,22,26,30	INTA	INTC
	INTB	INTD
	INTC	INTA
	INTD	INTB
3,7,11,15,19,23,27,31	INTA	INTD
	INTB	INTA
	INTC	INTB
	INTD	INTC

Note that the Requester ID of an Assert_INTx/Deassert_INTx Message will correspond to the Transmitter of the Message on that Link, and not necessarily to the original source of the interrupt.



IMPLEMENTATION NOTE

System Interrupt Mapping

Note that system software (including BIOS and operating system) needs to comprehend the remapping of legacy interrupts (INTx mechanism) in the entire topology of the system (including hierarchically connected Switches and subordinate PCI Express/PCI Bridges) to establish proper correlation between PCI Express device interrupt and associated interrupt resources in the system interrupt controller. The remapping described by Table 2-20 is applied hierarchically at every Switch. In addition, PCI Express/PCI and PCI/PCI Bridges perform a similar mapping function.



IMPLEMENTATION NOTE

Virtual Wire Mapping for INTx Interrupts From ARI Devices

The implied Device Number for an ARI Device is 0. When ARI-aware software (including BIOS and operating system) enables ARI Forwarding in the Downstream Port immediately above an ARI Device in order to access its Extended Functions, software must comprehend that the Downstream Port will use Device Number 0 for the virtual wire mappings of INTx interrupts coming from all Functions of the ARI Device. If non-ARI-aware software attempts to determine the virtual wire mappings for Extended Functions, it can come up with incorrect mappings by examining the traditional Device Number field and finding it to be non-0.

2.2.8.2. Power Management Messages

These Messages are used to support PCI Express power management, which is described in detail in Chapter 5. The following rules define the Power Management Messages:

- Table 2-21 defines the Power Management Messages.
- Power Management Messages do not include a data payload (TLP Type is Msg).
- The Length field is Reserved.
- With PM_Active_State_Nak Messages, the Function Number field in the Requester ID must contain the Function Number of the Downstream Port that sent the Message, or else 000b for compatibility with earlier revisions of this specification.
- With PME_TO_Ack Messages, the Function Number field in the Requester ID must be Reserved, or else for compatibility with earlier revisions of this specification must contain the Function Number of one of the Functions associated with the Upstream Port. Note that the Function Number field is a different size for non-ARI and ARI Requester IDs.

- Power Management Messages must use the default Traffic Class designator (TC0). Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

Table 2-21: Power Management Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
PM_Active_State_Nak	0001 0100	100	t	r	tr	r	Terminate at Receiver
PM_PME	0001 1000	000	All:				Sent Upstream by PME-requesting component. Propagates Upstream.
			r		tr	t	
			If PME supported:				
				t			
PME_Turn_Off	0001 1001	011	t	r		r	Broadcast Downstream
PME_TO_Ack	0001 1011	101	r	t		t	Sent Upstream by Upstream Port. See Section 5.3.3.2.1.
			(Note: Switch handling is special)				

2.2.8.3. Error Signaling Messages

Error Signaling Messages are used to signal errors that occur on specific transactions and errors that are not necessarily associated with a particular transaction. These Messages are initiated by the agent that detected the error.

- Table 2-22 defines the Error Signaling Messages.
- Error Signaling Messages do not include a data payload (TLP Type is Msg).
- The Length field is Reserved.
- With Error Signaling Messages, the Function Number field in the Requester ID must indicate which Function is signaling the error. Note that the Function Number field is a different size for non-ARI and ARI Requester IDs.
- Error Signaling Messages must use the default Traffic Class designator (TC0) Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

Table 2-22: Error Signaling Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
ERR_COR	0011 0000	000	r	t	tr	t	This Message is issued when the Function or Device detects a correctable error on the PCI Express interface.
ERR_NONFATAL	0011 0001	000	r	t	tr	t	This Message is issued when the Function or Device detects a Non-fatal, uncorrectable error on the PCI Express interface.
ERR_FATAL	0011 0011	000	r	t	tr	t	This Message is issued when the Function or Device detects a Fatal, uncorrectable error on the PCI Express interface.

The initiator of the Message is identified with the Requester ID of the Message header. The Root Complex translates these error Messages into platform level events. Refer to Section 6.2 for details on uses for these Messages.

2.2.8.4. Locked Transactions Support

The Unlock Message is used to support Lock Transaction sequences. Refer to Section 6.5 for details on Lock Transaction sequences. The following rules apply to the formation of the Unlock Message:

- Table 2-23 defines the Unlock Messages.
- The Unlock Message does not include a data payload (TLP Type is Msg).
- The Length field is Reserved.
- With Unlock Messages, the Function Number field in the Requester ID is Reserved.
- The Unlock Message must use the default Traffic Class designator (TC0) Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

Table 2-23: Unlock Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/ Comments
			RC	Ep	Sw	Br	
Unlock	0000 0000	011	t	r	tr	r	Unlock Completer

2.2.8.5. Slot Power Limit Support

This Message is used to convey a slot power limitation value from a Downstream Port (of a Root Complex or a Switch) to an Upstream Port of a component (with Endpoint, Switch, or PCI Express-PCI Bridge Functions) attached to the same Link.

- Table 2-24 defines the Set_Slot_Power_Limit Message.
- The Set_Slot_Power_Limit Message includes a 1 DW data payload (TLP Type is MsgD).
- The Set_Slot_Power_Limit Message must use the default Traffic Class designator (TC0) Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

Table 2-24: Set_Slot_Power_Limit Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
Set_Slot_Power_Limit	0101 0000	100	t	r	tr	r	Set Slot Power Limit in Upstream Port

The Set_Slot_Power_Limit Message includes a one DW data payload. The data payload is copied from the Slot Capabilities register of the Downstream Port and is written into the Device Capabilities register of the Upstream Port on the other side of the Link. Bits 1:0 of Byte 1 of the data payload map to the Slot Power Limit Scale field and bits 7:0 of Byte 0 map to the Slot Power Limit Value field. Bits 7:0 of Byte 3, 7:0 of Byte 2, and 7:2 of Byte 1 of the data payload must be set to all 0's by the Transmitter and ignored by the Receiver. This Message must be sent automatically by the Downstream Port (of a Root Complex or a Switch) when one of the following events occurs:

- On a Configuration Write to the Slot Capabilities register (see Section 7.8.9) when the Data Link Layer reports DL_Up status.
- Any time when a Link transitions from a non-DL_Up status to a DL_Up status (see Section 2.9.2) and the Auto Slot Power Limit Disable bit is Cleared in the Slot Control Register. This Transmission is optional if the Slot Capabilities register has not yet been initialized.

Comment [VS56]: ECN: Downstream Port containment (DPC)

The component on the other side of the Link (with Endpoint, Switch, or Bridge Functions) that receives Set_Slot_Power_Limit Message must copy the values in the data payload into the Device Capabilities register associated with the component's Upstream Port. PCI Express components that are targeted exclusively for integration on the system planar (e.g., system board) as well as components that are targeted for integration on a card/module where power consumption of the entire card/module is below the lowest power limit specified for the card/module form factor (as

defined in the corresponding form factor specification) are permitted to hardwire the value of all 0's in the Slot Power Limit Scale and Slot Power Limit Value fields of the Device Capabilities register, and are not required to copy the Set_Slot_Power limit payload into that register.

For more details on Power Limit control mechanism see Section 6.9.

2.2.8.6. Vendor_Defined Messages

The Vendor_Defined Messages allow expansion of PCI Express messaging capabilities, either as a general extension to the PCI Express Specification or a vendor-specific extension. ~~Such extensions are not covered specifically in this document, although future revisions of this specification may use this mechanism to define new Messages (see below)~~. This section defines the rules associated with these Messages generically.

Comment [VS57]: ECN: Lightweight Notification (LN) Protocol

- The Vendor_Defined Messages (see Table 2-25) use the header format shown in Figure 2-25.
 - The Requester ID is implementation specific. It is strongly recommended that the Requester ID field contain the value associated with the Requester.[15]
 - If the Route by ID routing is used, bytes 8 and 9 form a 16-bit field for the destination ID
 - otherwise these bytes are Reserved.
 - Bytes 10 and 11 form a 16-bit field for the Vendor ID, as defined by PCI-SIG®, of the vendor defining the Message.
 - Bytes 12 through 15 are available for vendor definition.

Comment [VS58]: Errata A3

Table 2-25: Vendor_Defined Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
Vendor_Defined Type 0	0111 1110	000, 010, 011, 100	See Note 1.				Triggers detection of UR by Completer if not implemented.
Vendor_Defined Type 1	0111 1111	000, 010, 011, 100	See Note 1.				Silently discarded by Completer if not implemented.

Note 1: Transmission by Endpoint/Root Complex/Bridge is implementation specific. Switches must forward received Messages using Routing[2:0] field values of 000b, 010b, and 011b.

[15] ACS Source Validation (see Section 6.12.1.1) checks the Requester ID on all Requests, including Vendor_Defined Messages. This validation depends on the Requester ID properly identifying the Requester.

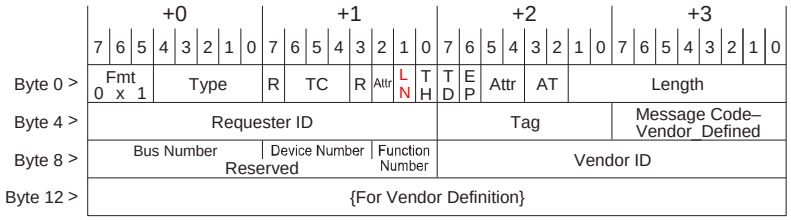


Figure 2-25: Header for Vendor-Defined Messages OM13775B

- A data payload may be included with either type of Vendor_Defined Message (TLP type is Msg if no data payload is included and MsgD if a data payload is included)
- For both types of Vendor_Defined Messages, the Attr[1:0] and Attr[2] fields are not Reserved.
- Messages defined by different vendors or by PCI-SIG are distinguished by the value in the Vendor ID field.
 - The further differentiation of Messages defined by a particular vendor is beyond the scope of this document.
 - Support for Messages defined by a particular vendor is implementation specific, and beyond the scope of this document.
- Completers silently discard Vendor_Defined Type 1 Messages which they are not designed to receive – this is not an error condition.
- Completers handle the receipt of an unsupported Vendor_Defined Type 0 Message as an Unsupported Request, and the error is reported according to Section 6.2.

PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0 defines additional requirements for Vendor_Defined Messages that are designed to be interoperable with PCI-X Device ID Messages. This includes restrictions on the contents of the Tag[7:0] field and the Length[9:0] field as well as specific use of Bytes 12 through 15 of the message header. Vendor_Defined Messages intended for use solely within a PCI Express environment (i.e., not intended to address targets behind a PCI Express to PCI/PCI-X Bridge) are not subject to the additional rules. Refer to *PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0* for details.

2.2.8.6.1. PCI-SIG-Defined VDMs

Comment [VS59]: ECN: Lightweight Notification (LN) Protocol

PCI-SIG-Defined VDMs are Vendor-Defined Type 1 Messages that use the PCI-SIG® Vendor ID (0001h). As a Vendor-Defined Type 1 Message, each is silently discarded by a Completer if the Completer does not implement it.

Beyond the rules for other Vendor-Defined Type 1 Messages, the following rules apply to the formation of the PCI-SIG-Defined VDMs:

- PCI-SIG-Defined VDMs use the Header format shown in Figure 2-26.
- The Requester ID field must contain the value associated with the Requester.
- The Message Code must be 01111111b.

- The Vendor ID must be 0001h, which is assigned to the PCI-SIG.
- The Subtype field distinguishes the specific PCI-SIG-Defined VDMs. See Appendix F for a list of PCI-SIG-Defined VDMs.

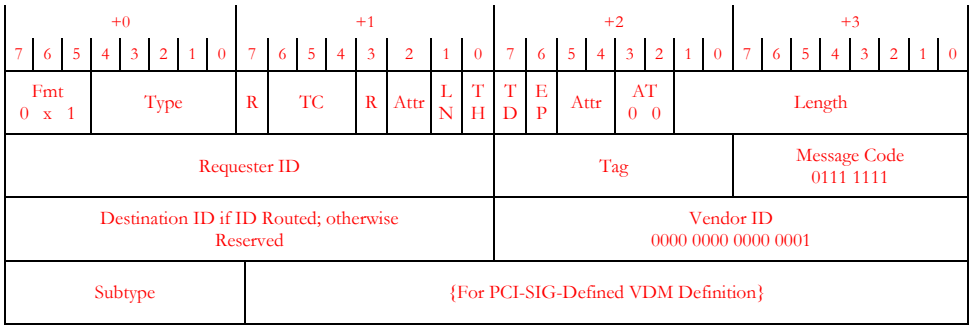


Figure 2-26: Header for PCI-SIG-Defined VDMs

2.2.8.6.2. LN Messages

Comment [VS60]: ECN: Lightweight Notification (LN) Protocol

LN Messages

LN protocol (see 6.21) defines LN Messages, which are PCI-SIG-Defined VDMs. The payload of each Message generally contains the 64-bit Address of a registered cacheline that has been updated or evicted. The single 64-bit address format is used both with 64- and 32-bit addresses. Since each LN Message is a Vendor-Defined Type 1 Message, a Completer that receives a properly formed LN Message is required to silently discard it if the Completer doesn't recognize the Message.

An LN Message can be directed to a single Endpoint using ID-based routing, or broadcast to all devices below a given Root Port. Whether a broadcast LN Message is sent to all Root Ports in the RC is implementation specific.

Beyond the rules for other PCI-SIG-Defined VDMs, the following rules apply to the formation of LN Messages:

- Table 2-27 and Figure 2-27 define the LN Messages.
- Each Message must include a 2-DWORD data payload.
- The Fmt field must be 011b (4 DW Header, with data).
- The TLP Type must be MsgD.
- The Length field must be 2.
- The TC[2:0] field must be 000b.
- Attr[2], the ID-Based Ordering (IDO) bit, is not Reserved.
- Attr[1], the Relaxed Ordering (RO) bit, is not Reserved.
- Attr[0], the No Snoop bit, is Reserved.

- Note: as with all Messages, the LN bit is Reserved. In contrast, the LN bit must be Set for LN Reads, LN Writes, and LN Completions.
- The Tag field is Reserved.
- If the LN Message is the broadcast version, the Destination ID field is Reserved.
- The Subtype field must be 00h.
- If the cacheline size in effect for the system is 128 bytes, bit 6 in the Cacheline Address must be Clear. For an LNR receiving an LN Message, if the LNR CLS bit in the LNR Control register is Set, configuring the LNR for 128-byte cachelines, the LNR must ignore the value of bit 6 in the Cacheline Address.
- The Notification Reason (NR) field is encoded as shown in Table 2-26, indicating the specific reason that the LN Message was sent. These encodings apply to both the directed and broadcast versions of LN Messages.

Table 2-26: Notification Reason (NR) Field Encodings

NR Coding (b)	Description
00	LN Message was sent due to a cacheline update.
01	LN Message was sent due to the eviction of a single cacheline.
10	LN Message was sent due to the eviction of all cachelines registered to this Function. For this case, the Cacheline Address is Reserved.
11	Reserved

Table 2-27: LN Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
LN Message (directed)	0111 1111	010	t	r	tr	r	RC directs to a single Endpoint.
LN Message (broadcast)	0111 1111	011	t	r	tr	r	RC broadcasts to all devices under a given Root Port.

The format of the LN Message is shown in Figure 2-27 below.

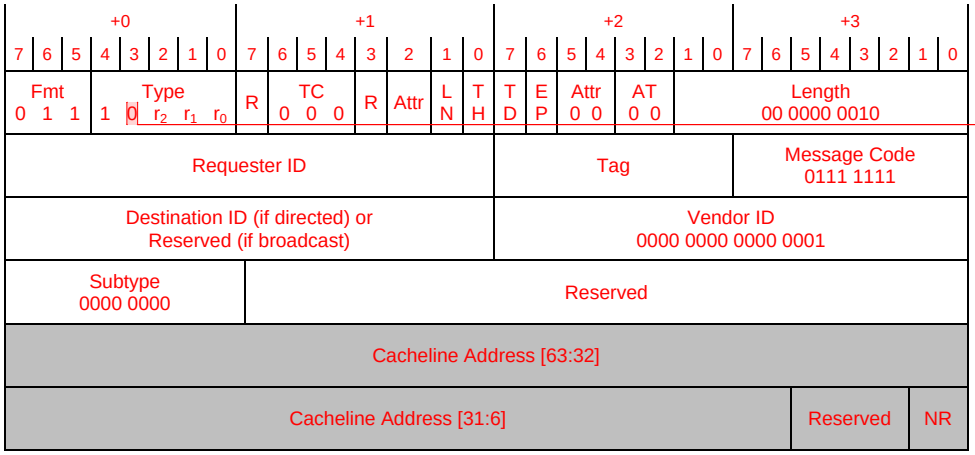


Figure 2-27: LN Message

Comment [VS61]: Type 0 1 r2 r1 r0 chaged to 1 0 r2 r1 r0 per Errata B30 (LN ECN)

2.2.8.6.3. Device Readiness Status (DRS) Message

Comment [VS62]: D-ECN-Readiness Notifications

The Device Readiness Status (DRS) protocol (see Section 6.23.1) uses the PCI-SIG-Defined VDM mechanism (see Section 2.2.8.6.1). The DRS Message is a PCI-SIG-Defined VDM (Vendor-Defined Type 1 Message) with no payload.

Beyond the rules for other PCI-SIG-Defined VDMs, the following rules apply to the formation of DRS Messages:

- Table 2-28 and Figure 2-28 illustrate and define the DRS Message.
- The TLP Type must be Msg.
- The TC[2:0] field must be 000b.
- The Attr[2:0] field is Reserved.
- The Tag field is Reserved.
- The Message Routing field must be set to 100b – Local – Terminate at Receiver.

Receivers may optionally check for violations of these rules (but must not check reserved bits). These checks are independently optional (see Section 6.2.3.4). If a Receiver implementing these checks determines that a TLP violates these rules, the TLP is a Malformed TLP.

Comment [VS63]: RC_ECN_RN_27_Aug_2013

- If checked, this is a reported error associated with the Receiving Port (see Section 6.2).

Comment [VS64]: RC_ECN_RN_27_Aug_2013

Table 2-28: DRS Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
DRS Message	0111 1111	100	r	t	tr		Device Readiness Status

The format the DRS Message is shown in Figure 2-28 below.

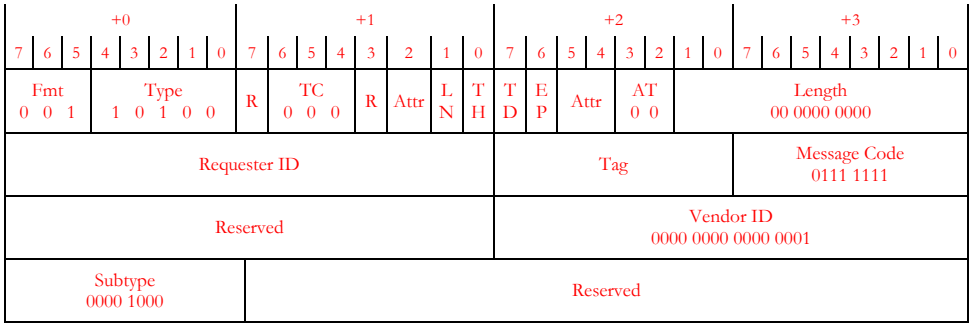


Figure 2-28: DRS Message

Comment [VS65]: Bits [5:4] changed per RC_ECN_RN_27_Aug_2013

2.2.8.6.4. Function Readiness Status (FRS) Message

The Function Readiness Status (FRS) protocol (see Section 6.23.2) uses the PCI-SIG-Defined VDM mechanism (see Section 2.2.8.6.1). The FRS message is a PCI-SIG-Defined VDM (Vendor-Defined Type 1 Message) with no payload.

Comment [VS66]: Para updated per RC_ECN_RN_27_Aug_2013

Beyond the rules for other PCI-SIG-Defined VDMs, the following rules apply to the formation of FRS Messages:

- Table 2-29 and Figure 2-29 illustrate and define the FRS Message.
- The TLP Type must be Msg.
- The TC[2:0] field must be 000b.
- The Attr[2:0] field is Reserved.
- The Tag field is Reserved.
- The FRS Reason[3:0] field indicates why the FRS Message was generated:
 - **0001b: DRS Message Received** - The Downstream Port indicated by the Message Requester ID received a DRS Message and has the DRS Signaling Control field in the Link Control Register set to DRS to FRS Signaling Enabled
 - **0010b: $D3_{hot}$ to D0 Transition Completed** - A $D3_{hot}$ to D0 transition has completed, and the Function indicated by the Message Requester ID is now Configuration-Ready

Formatted: No underline, Font color: Auto, Subscript

Formatted: No underline, Font color: Auto, Subscript

and has returned to the $D0_{uninitialized}$ or $D0_{active}$ state depending on the setting of the No_Soft_Reset bit (see *PCI Bus Power Management Interface Specification* Section 3.2.4 and Section 5.4.1)

- **0011b: FLR Completed** - An FLR has completed, and the Function indicated by the Message Requester ID is now Configuration-Ready
- **1000b: VF Enabled** – The Message Requester ID indicates a PF - All VFs associated with that PF are now Configuration-Ready
- **1001b: VF Disabled** – The Message Requester ID indicates a PF - All VFs associated with that PF have been disabled and the SR-IOV data structures in that PF may now be accessed.
- All other values Reserved

❑ The Message Routing field must be set to 000b – Routed to Root Complex

Receivers may optionally check for violations of these rules (but must not check reserved bits). These checks are independently optional (see Section 6.2.3.4). If a Receiver implementing these checks determines that a TLP violates these rules, the TLP is a Malformed TLP.

❑ If checked, this is a reported error associated with the Receiving Port (see Section 6.2).

Comment [VS67]: Bullet point updated per RC_ECN_RN_27_Aug_2013

Comment [VS68]: RC_ECN_RN_27_Aug_2013

Comment [VS69]: RC_ECN_RN_27_Aug_2013

Table 2-29: FRS Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
FRS Message	0111 1111	000	r	t	tr		Function Readiness Status

The format the FRS Message is shown in Figure 2-29 below.

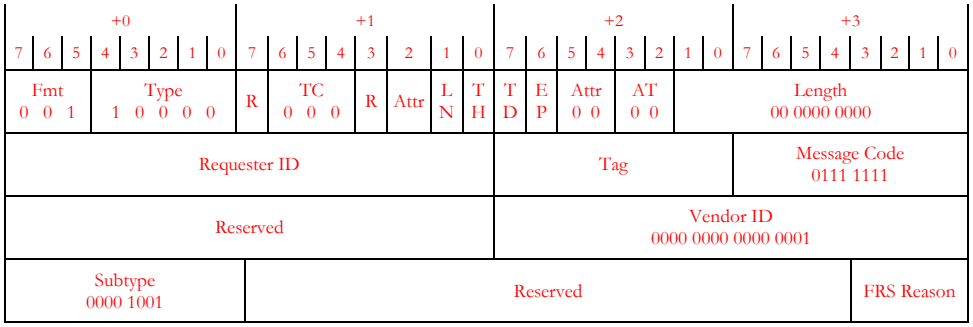


Figure 2-29: FRS Message

2.2.8.7. Ignored Messages

The messages listed in Table 2-30 were previously used for a mechanism (Hot-Plug Signaling) that is no longer supported. Transmitters are strongly encouraged not to transmit these messages, but if message transmission is implemented, it must conform to the requirements of the 1.0a version of this specification.

Receivers are strongly encouraged to ignore receipt of these messages, but are allowed to process these messages in conformance with the requirements of 1.0a version of this specification.

Ignored messages listed in Table 2-30 are handled by the Receiver as follows:

- The Physical and Data Link Layers must handle these messages identical to handling any other TLP.
- The Transaction Layer must account for flow control credit but take no other action in response to these messages.

Table 2-30: Ignored Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
			RC	Ep	Sw	Br	
Ignored Message	0100 0001	100					
Ignored Message	0100 0011	100					
Ignored Message	0100 0000	100					
Ignored Message	0100 0101	100					
Ignored Message	0100 0111	100					
Ignored Message	0100 0100	100					
Ignored Message	0100 1000	100					

2.2.8.8. Latency Tolerance Reporting (LTR) Message

The LTR Message is optionally used to report device behaviors regarding its tolerance of Read/Write service latencies. Refer to Section 6.18 for details on LTR. The following rules apply to the formation of the LTR Message:

- Table 2-31 defines the LTR Message.
- The LTR Message does not include a data payload (the TLP Type is Msg).
- The Length field is Reserved.
- The LTR Message must use the default Traffic Class designator (TC0). Receivers that implement LTR support must check for violations of this rule. If a Receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

Table 2-31: LTR Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support[16]				Description/Comments
			RC	Ep	Sw	Br	
LTR	0001 0000	100	r	t	tr		Latency Tolerance Reporting

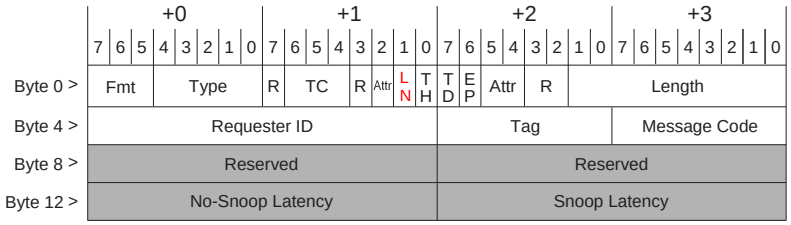


Figure 2-30: LTR Message

2.2.8.9. Optimized Buffer Flush/Fill (OBFF) Message

The OBFF Message is optionally used to report platform central resource states to Endpoints. This mechanism is described in detail in Section 6.19.

The following rules apply to the formation of the OBFF Message:

- Table 2-32 defines the OBFF Message.
- The OBFF Message does not include a data payload (TLP Type is Msg).
- The Length field is Reserved.
- The Requester ID must be set to the Transmitting Port's ID.
- The OBFF Message must use the default Traffic Class designator (TC0). Receivers that implement OBFF support must check for violations of this rule. If a Receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.

This is a reported error associated with the Receiving Port (see Section 6.2).

[16] Support for LTR is optional. Functions that support LTR must implement the reporting and enable mechanisms described in Chapter 7.

Table 2-32: OBFF Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support[1]				~~Req ID~~	Description/Comments
			RC	Ep	Sw	Br		
OBFF	0001 0010	100	t	r	tr		~~BD~~	Optimized Buffer Flush/Fill

Note 1: Support for OBFF is optional. Functions that support OBFF must implement the reporting and enable mechanisms described in Chapter 7.

Comment [VS70]: Per Steve Glaser, This was removed in other messages.

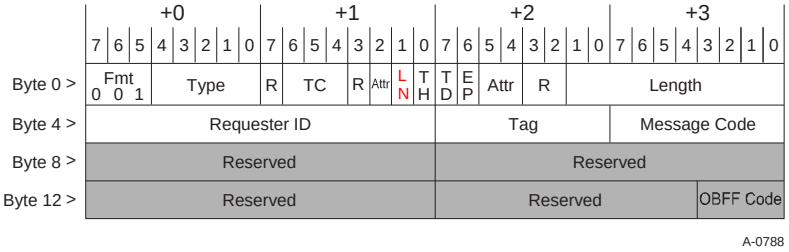


Figure 2-31: OBFF Message

2.2.8.10. Precision Time Measurement (PTM) Messages

Comment [VS71]: ECN: Precision Time Measurement

Table 2-33 defines the Precision Time Measurement (PTM) Messages.

- The PTM Request and PTM Response Messages must use a TLP Type of Msg, and must not include a data payload. The Length field is reserved.
 - Figure 2-32 illustrates the format of the PTM Request and Response Messages.
- The PTM ResponseD Message must use a TLP Type of MsgD, and must include a 64 bit PTM Master Time field in bytes 8 through 15 of the TLP header and a 1 DW data payload containing the 32 bit Propagation Delay field.
 - Figure 2-33 illustrates the format of the PTM ResponseD Message.
 - Refer to section 6.22.3.2 for details regarding how to populate the PTM ResponseD Message.
- The Requester ID must be set to the Transmitting Port's ID.
- A PTM dialog is defined as a matched pair of messages consisting of a PTM Request and the corresponding PTM Response or PTM ResponseD message.
- The PTM Messages must use the default Traffic Class designator (TC0). Receivers implementing PTM must check for violations of this rule. If a receiver determines that a TLP violates this rule, it must handle the TLP as a Malformed TLP.
 - This is a reported error associated with the Receiving Port (see Section 6.2).

Table 2-33: Precision Time Measurement Messages

Name	TLP Type	Code[7:0] (b)	Routing r[2:0] (b)	Support				Description/Comments
				RC	EP	SW	BR	
PTM Request	Msg	0101 0010	100	r	t	tr		Initiates PTM dialog
PTM Response	Msg	0101 0011	100	t	r	tr		Completes current PTM dialog – does not carry timing information
PTM ResponseD	MsgD	0101 0011	100	t	r	tr		Completes current PTM dialog - carries timing information

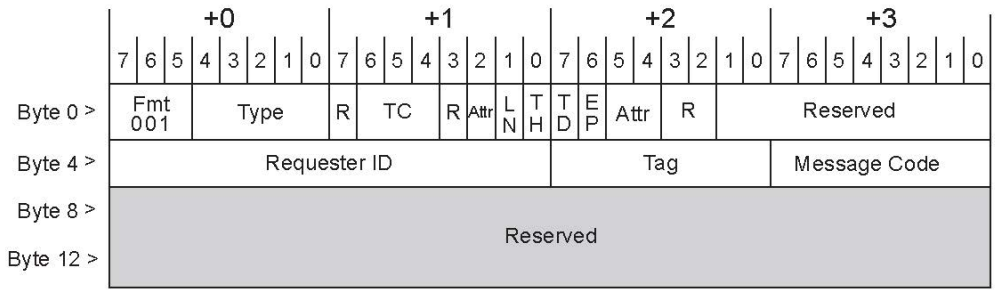


Figure 2-32: PTM Request/Response Message

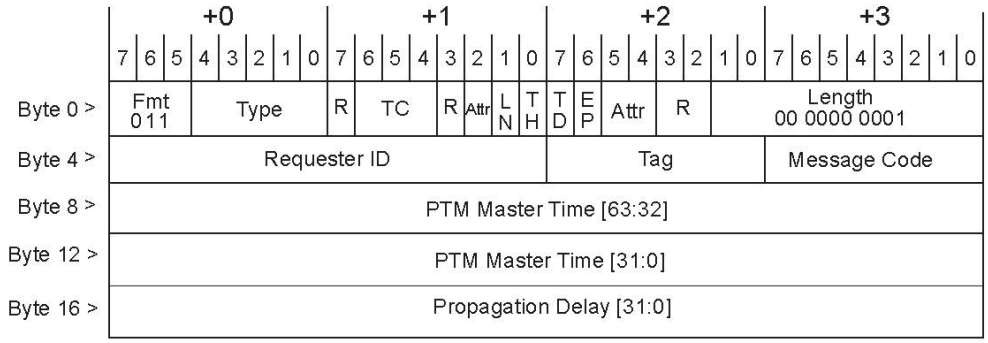


Figure 2-33: PTM ResponseD Message (4 DW header and 1 DW payload)

2.2.9. Completion Rules

All Read, Non-Posted Write, and AtomicOp Requests require Completion. Completions include a Completion header that, for some types of Completions, will be followed by some number of DW of data. The rules for each of the fields of the Completion header are defined in the following sections.

- Completions route by ID, and use a 3 DW header.
 - Note that the routing ID fields correspond directly to the Requester ID supplied with the corresponding Request. Thus for Completions these fields will be referred to collectively as the Requester ID instead of the distinct fields used generically for ID routing.
- In addition to the header fields included in all TLPs and the ID routing fields, Completions contain the following additional fields (see Figure 2-34):
 - Completer ID[15:0] – Identifies the Completer – described in detail below
 - Completion Status[2:0] – Indicates the status for a Completion (see Table 2-34)
 - Rules for determining the value in the Completion Status[2:0] field are in Section 2.3.1.
 - BCM – Byte Count Modified – this bit must not set by PCI Express Completers, and may only be set by PCI-X completers
 - Byte Count[11:0] – The remaining byte count for Request
 - The Byte Count value is specified as a binary number, with 0000 0000 0001b indicating 1 byte, 1111 1111 1111b indicating 4095 bytes, and 0000 0000 0000b indicating 4096 bytes.
 - For Memory Read Completions, Byte Count[11:0] is set according to the rules in Section 2.3.1.1.

- For AtomicOp Completions, the Byte Count value must equal the associated AtomicOp operand size in bytes.
- For all other types of Completions, the Byte Count value must be 4.

- Tag[7:0] – in combination with the Requester ID field, corresponds to the Transaction ID
- Lower Address[6:0] – lower byte address for starting byte of Completion
 - For Memory Read Completions, the value in this field is the byte address for the first enabled byte of data returned with the Completion (see the rules in Section 2.3.1.1).
 - For AtomicOp Completions, the Lower Address field is Reserved.
 - This field is set to all 0's for all remaining types of Completions. Receivers may optionally check for violations of this rule. See Section 2.3.2, second bullet, for details.

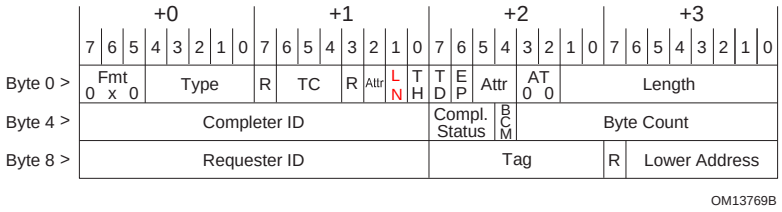


Figure 2-34: Completion Header Format

Table 2-34: Completion Status Field Values

Completion Status[2:0] Field Value (b)	Completion Status
000	Successful Completion (SC)
001	Unsupported Request (UR)
010	Configuration Request Retry Status (CRS)
100	Completer Abort (CA)
all others	Reserved

- The Completer ID[15:0] is a 16-bit value that is unique for every PCI Express Function within a Hierarchy (see Figure 2-35 and Figure 2-36)

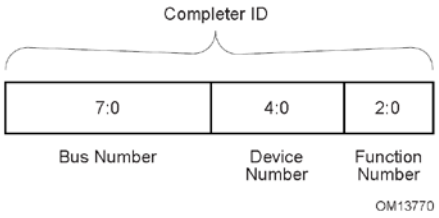


Figure 2-35: (Non-ARI) Completer ID

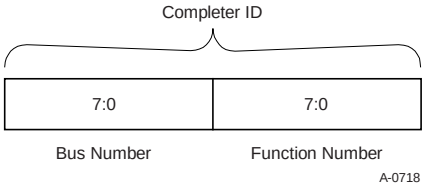


Figure 2-36: ARI Completer ID

- Functions must capture the Bus and Device Numbers[17] supplied with all Type 0 Configuration Write Requests completed by the Function, and supply these numbers in the Bus and Device Number fields of the Completer ID[18] for all Completions generated by the Device/Function.
 - If a Function must generate a Completion prior to the initial device Configuration Write Request, 0's must be entered into the Bus Number and Device Number fields
 - Note that Bus Number and Device Number may be changed at run time, and so it is necessary to re-capture this information with each and every Configuration Write Request.
 - Exception: The assignment of Bus Numbers to the Devices within a Root Complex may be done in an implementation specific way.
- In some cases, a Completion with the UR status may be generated by a multi-Function device without associating the Completion with a specific Function within the device – in this case, the Function Number field[19] is Reserved.
 - Example: A multi-Function device receives a Read Request which does not target any resource associated with any of the Functions of the device – the device generates a Completion with UR status and sets a value of all 0's in the Function Number field of the Completer ID.

[17] With ARI Devices, Functions are only required to capture the Bus Number. ARI Devices are permitted to retain the captured Bus Number on either a per-Device or a per-Function basis. See Section 2.2.6.2.

[18] An ARI Completer ID does not contain a Device Number field. See Section 2.2.4.2.

[19] Note: with an ARI Completer ID, the Function Number field is 8 bits.

- Completion headers must supply the same values for the Requester ID, Tag, and Traffic Class as were supplied in the header of the corresponding Request.
- Completion headers must supply the same values for the Attribute as were supplied in the header of the corresponding Request, except as explicitly allowed when IDO is used (see Section 2.2.6.4).
- If the Completer is an LN Completer and the targeted memory region supports registrations, the following rules apply; otherwise the value of the LN Bit must be 0b.
 - If the Completion Status is Successful Completion and the associated Request was an LN Read, the LN bit must be 1b.
 - Otherwise the LN bit must be 0b.
- The TH bit is reserved for Completions.
- AT[1:0] must be 00b. Receivers are not required or encouraged to check this.
- The Completion ID field is not meaningful prior to the software initialization and configuration of the completing device (using at least one Configuration Write Request), and for this case the Requester must ignore the value returned in the Completer ID field.
- A Completion including data must specify the actual amount of data returned in that Completion, and must include the amount of data specified.
 - It is a TLP formation error to include more or less data than specified in the Length field, and the resulting TLP is a Malformed TLP.

 Note: This is simply a specific case of the general rule requiring the TLP data payload length to match the value in the Length field.

Comment [VS72]: ECN: Lightweight Notification (LN) Protocol

2.2.10. TLP Prefix Rules

The following rules apply to any TLP that contains a TLP Prefix:

- For any TLP, a value of 100b in the Fmt[2:0] field in byte 0 of the TLP indicates the presence of a TLP Prefix and the Type[4] bit indicates the type of TLP Prefix
 - A value of 0b in the Type[4] bit indicates the presence of a Local TLP Prefix
 - A value of 1b in the Type[4] bit indicates the presence of an End-End TLP Prefix
- The format for bytes 1 through 3 of a TLP Prefix are defined by its TLP Prefix type
- A TLP that contains a TLP Prefix must have an underlying TLP Header. A received TLP that violates this rule is handled as a Malformed TLP. This is a reported error associated with the Receiving Port (see Section 6.2)
- It is permitted for a TLP to contain more than one TLP Prefix of any type
 - When a combination of Local and End-End TLP Prefixes are present in TLP, it is required that all the Local TLP Prefixes precede any End-End TLP Prefixes. A received TLP that violates this rule is handled as a Malformed TLP. This is a reported error associated with the Receiving Port (see Section 6.2).

- The size of each TLP Prefix is 1 DW. A TLP Prefix may be repeated to provide space for additional data.
- If the value in the Fmt and Type field indicates the presence of a Local TLP Prefix, handle according to the Local TLP Prefix handling (see Section 2.2.10.1)
- If the value in the Fmt and Type field indicates the presence of an End-End TLP Prefix, handle according to the End-End TLP Prefix handling (see Section 2.2.10.2).

2.2.10.1. Local TLP Prefix Processing

The following rules apply to Local TLP Prefixes:

- Local TLP Prefix types are determined using the L[3:0] sub-field of the Type field
 - Type[4] must be 0b
 - Local TLP Prefix L[3:0] values are defined in Table 2-35

Table 2-35: Local TLP Prefix Types

Local TLP Prefix Type	L[3:0] (b)	Description
MR-IOV	0000	**MR-IOV TLP Prefix** – Refer to the *Multi-Root I/O Virtualization and Sharing* specification for details.
VendPrefixL0	1110	**Vendor Defined Local TLP Prefix** – Refer to Section 2.2.10.1.1 for further details.
VendPrefixL1	1111	**Vendor Defined Local TLP Prefix** – Refer to Section 2.2.10.1.1 for further details.
		All other encodings are Reserved.

- The size, routing, and flow control rules are specific to each Local TLP Prefix type.
- It is an error to receive a TLP with a Local TLP Prefix type not supported by the Receiver. If the Extended Fmt Field Supported bit is Set, TLPs in violation of this rule are handled as a Malformed TLP unless explicitly stated differently in another specification. This is a reported error associated with the Receiving Port (see Section 6.2). If the Extended Fmt Field Supported bit is Clear, behavior is device specific.
- No Local TLP Prefixes are protected by ECRC even if the underlying TLP is protected by ECRC.

2.2.10.1.1. Vendor Defined Local TLP Prefix

As described in Table 2-35, Types VendPrefixL0 and VendPrefixL1 are Reserved for use as Vendor Defined Local TLP Prefixes. To maximize interoperability and flexibility the following rules are applied to such prefixes:

- Components must not send TLPs containing Vendor Defined Local TLP Prefixes unless this has been explicitly enabled (using vendor specific mechanisms).
- Components that support any usage of Vendor Defined Local TLP Prefixes must support the 3-bit definition of the Fmt field and have the Extended Fmt Field Supported bit Set (see Section 7.8.15).
- It is recommended that components be configurable (using vendor specific mechanisms) so that all vendor defined prefixes can be sent using either of the two Vendor Defined Local TLP Prefix encodings. Such configuration need not be symmetric (for example each end of a Link could transmit the same Prefix using a different encoding).

2.2.10.2. End-End TLP Prefix Processing

The following rules apply to End-End TLP Prefixes

- End-End TLP Prefix types are determined using the E[3:0] sub-field of the Type field
 - Type[4] must be 1b
 - End-End TLP Prefix E[3:0] values are defined in Table 2-36

Table 2-36: End-End TLP Prefix Types

End-End TLP Prefix Type	E[3:0] (b)	Description
ExtTPH	0000	**Extended TPH** – Refer to Section 6.17 for further details.
PASID	0001	**PASID** – Refer to Section 6.20 for further details.
VendPrefixE0	1110	**Vendor Defined End-End TLP Prefix** – Refer to Section 2.2.10.2.1 for further details.
VendPrefixE1	1111	**Vendor Defined End-End TLP Prefix** – Refer to Section 2.2.10.2.1 for further details.
		All other encodings are Reserved.

Comment [VS73]: ECN:PASID

- The maximum number of End-End TLP Prefixes permitted in a TLP is 4:
 - A Receiver supporting TLP Prefixes must check this rule. If a Receiver determines that a TLP violates this rule, the TLP is a Malformed TLP. This is a reported error associated with the Receiving Port (see Section 6.2).
- The presence of an End-End TLP Prefix does not alter the routing of a TLP. TLPs are routed based on the routing rules covered in Section 2.2.4.
- Functions indicate how many End-End TLP Prefixes they support by the Max End-End TLP Prefixes field in the Device Capabilities 2 register (see Section 7.8.15).

- For Root Ports, the Max End-End TLP Prefixes field is permitted to return a value indicating support for fewer End-End TLP Prefixes than what the Root Port hardware actually implements; however, the error handling semantics must still be based on the value contained in the field. TLPs received that contain more End-End TLP Prefixes than are supported by the Root Port must be handled as follows. It is recommended that Requests be handled as Unsupported Requests, but otherwise they must be handled as Malformed TLPs. It is recommended that Completions be handled as Unexpected Completions, but otherwise they must be handled as Malformed TLPs. For TLPs received by the Ingress Port, this is a reported error associated with the Ingress Port. For TLPs received internally to be transmitted out the Egress Port, this is a reported error associated with the Egress Port. See Section 6.2.
- For all other Function types, TLPs received that contain more End-End TLP Prefixes than are supported by a Function must be handled as Malformed TLPs. This is a reported error associated with the Receiving Port (see Section 6.2).

AER logging (if supported) occurs as specified in Section 6.2.4.4.

- ❑ Switches must support forwarding of TLPs with up to 4 End-End TLP Prefixes if the End-End TLP Prefix Supported bit is Set.
- ❑ Different Root Ports with the End-End TLP Prefix Supported bit Set are permitted to report different values for Max End-End TLP Prefixes.
- ❑ All End-End TLP Prefixes are protected by ECRC if the underlying TLP is protected by ECRC.
- ❑ It is an error to receive a TLP with an End-End TLP Prefix by a Receiver that does not support End-End TLP Prefixes. A TLP in violation of this rule is handled as a Malformed TLP. This is a reported error associated with the Receiving Port (see Section 6.2).

Comment [VS74]: Errata B24 (2.1 Errata B68)

- ❑ Software should ensure that TLPs containing End-End TLP Prefixes are not sent to components that do not support them. Components where the Extended Fmt Field Supported bit is Clear may misinterpret TLPs containing TLP Prefixes.
- ❑ If one Function of an Upstream Port has the End-End TLP Prefix Supported bit Set, all Functions of that Upstream Port must handle the receipt of a Request addressed to them that contains an unsupported End-End TLP Prefix type as an Unsupported Request. This is a reported error associated with the Receiving Port (see Section 6.2).
- ❑ If one Function of an Upstream Port has the End-End TLP Prefix Supported bit Set, all Functions of that Upstream Port must handle the receipt of a Completion addressed to them that contains an unsupported End-End TLP Prefix type as an Unexpected Completion. This is a reported error associated with the Receiving Port (see Section 6.2).
- ❑ For routing elements, the End-End TLP Prefix Blocking bit in each Egress Port determines whether TLPs containing End-End TLP Prefixes can be transmitted via that Egress Port (see Section 7.8.16). If forwarding is blocked the entire TLP is dropped and a TLP Prefix Blocked Error is reported. If the blocked TLP is a Non-Posted Request, the Egress Port returns a Completion with Unsupport Request Completion Status. The TLP Prefix Blocked Error is a reported error associated with the Egress Port (see Section 6.2).

- For routing elements where Multicast is enabled (see Section 6.14). End-End TLP Prefixes are replicated in all Multicast copies of a TLP. TLP Prefix Egress Blocking of Multicast packets is performed independently at each Egress Port.

2.2.10.2.1. Vendor Defined End-End TLP Prefix

As described in Table 2-36, Types VendPrefixE0 and VendPrefixE1 are Reserved for use as Vendor Defined End-End TLP Prefixes. To maximize interoperability and flexibility the following rules are applied to such prefixes:

- Components must not send TLPs containing Vendor Defined End-End TLP Prefixes unless this has been explicitly enabled (using vendor specific mechanisms).
- It is recommended that components be configurable (using vendor specific mechanisms) to use either of the two Vendor Defined End-End TLP Prefix encodings. Doing so allows two different Vendor Defined End-End TLP Prefixes to be in use simultaneously within a single PCI Express topology while not requiring that every source understand the ultimate destination of every TLP it sends.

2.2.10.2.2. Root Ports with End-End TLP Prefix Supported

Support for peer-to-peer routing of TLPs containing End-End TLP Prefixes between Root Ports is optional and implementation dependent. If an RC supports End-End TLP Prefix routing capability between two or more Root Ports, it must indicate that capability in each associated Root Port via the End-End TLP Prefix Supported bit in the Device Capabilities 2 register.

An RC is not required to support End-End TLP Prefix routing between all pairs of Root Ports that have the End-End TLP Prefix Supported bit Set. A Request with End-End TLP Prefixes that would require routing between unsupported pairs of Root Ports must be handled as an Unsupported Request (UR). A Completion with End-End TLP Prefixes that would require routing between unsupported pairs of Root Ports must be handled as an Unexpected Completion (UC). In both cases, this error is reported by the "sending" Port.

The End-End TLP Prefix Supported bit must be Set for any Root Port that supports forwarding of TLPs with End-End TLP Prefixes initiated by host software or Root Complex Integrated Endpoints. The End-End TLP Prefix Supported bit must be Set for any Root Ports that support forwarding of TLPs with End-End TLP Prefixes received on their Ingress Port to Root Complex Integrated Endpoints.

Different Root Ports with the End-End TLP Prefix Supported bit Set are permitted to report different values for Max End-End TLP Prefixes.

An RC that splits a TLP into smaller TLPs when performing peer-to-peer routing between Root Ports must replicate the original TLP's End-End TLP Prefixes in each of the smaller TLPs (see Section 1.3.1).

2.3. Handling of Received TLPs

This section describes how all Received TLPs are handled when they are delivered to the Receive Transaction Layer from the Receive Data Link Layer, after the Data Link Layer has validated the integrity of the received TLP. The rules are diagramed in the flowchart shown in Figure 2-37.

- Values in Reserved fields must be ignored by the Receiver.
- If the value in the Fmt field indicates the presence of at least one TLP Prefix:
 - Detect if additional TLP Prefixes are present in the header by checking the Fmt field in the first byte of subsequent DWORDs until the Fmt field does not match that of a TLP Prefix.
 - Handle all received TLP Prefixes according to TLP Prefix Handling Rules (see Section 2.2.10).
- If the Extended Fmt Field Supported bit is Set, Received TLPs which use encodings of Fmt and Type that are Reserved are Malformed TLPs (see Table 2-1 and Table 2-3).
 - This is a reported error associated with the Receiving Port (see Section 6.2).
- If the Extended Fmt Field Supported bit is Clear, processing of Received TLPs that have Fmt[2] Set is undefined.[20]
- All Received TLPs with Fmt[2] Clear and which use undefined Type field values are Malformed TLPs.

 This is a reported error associated with the Receiving Port (see Section 6.2).
- All Received Malformed TLPs must be discarded.
 - Received Malformed TLPs that are ambiguous with respect to which buffer to release or are mapped to an uninitialized Virtual Channel must be discarded without updating Receiver Flow Control information.
 - All other Received Malformed TLPs must be discarded, optionally not updating Receiver Flow Control information.
- Otherwise, update Receiver Flow Control tracking information (see Section 2.6)
- If the value in the Type field indicates the TLP is a Request, handle according to Request Handling Rules, otherwise, the TLP is a Completion – handle according to Completion Handling Rules (following sections)

[20] An earlier version of this specification reserved the bit now defined for Fmt[2].

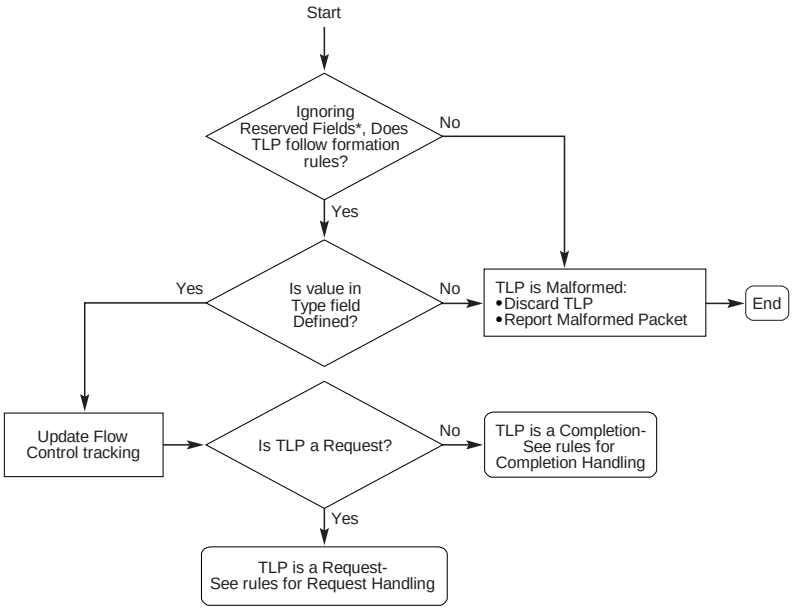


Figure 2-37: Flowchart for Handling of Received TLPs

Switches must process both TLPs which address resources within the Switch as well as TLPs which address resources residing outside the Switch. Switches handle all TLPs which address internal resources of the Switch according to the rules above. TLPs which pass through the Switch, or which address the Switch as well as passing through it, are handled according to the following rules (see Figure 2-38):

- If the value in the Type field indicates the TLP is not a Msg or MsgD Request, the TLP must be routed according to the routing mechanism used (see Sections 2.2.4.1 and 2.2.4.2).
- Switches route Completions using the information in the Requester ID field of the Completion.
- If the value in the Type field indicates the TLP is a Msg or MsgD Request, route the Request according to the routing mechanism indicated in the r[2:0] sub-field of the Type field
 - If the value in r[2:0] indicates the Msg/MsgD is routed to the Root Complex (000b), the Switch must route the Msg/MsgD to the Upstream Port of the Switch
 - It is an error to receive a Msg/MsgD Request specifying 000b routing at the Upstream Port of a Switch. Switches may check for violations of this rule – TLPs in violation are Malformed TLPs. If checked, this is a reported error associated with the Receiving Port (see Section 6.2).
 - If the value in r[2:0] indicates the Msg/MsgD is routed by address (001b), the Switch must route the Msg/MsgD in the same way it would route a Memory Request by address

- If the value in r[2:0] indicates the Msg/MsgD is routed by ID (010b), the Switch must route the Msg/MsgD in the same way it would route a Completion by ID
- If the value in r[2:0] indicates the Msg/MsgD is a broadcast from the Root Complex (011b), the Switch must route the Msg/MsgD to all Downstream Ports of the Switch
 - It is an error to receive a Msg/MsgD Request specifying 011b routing at the Downstream Port of a Switch. Switches may check for violations of this rule – TLPs in violation are Malformed TLPs. If checked, this is a reported error associated with the Receiving Port (see Section 6.2).
- If the value in r[2:0] indicates the Msg/MsgD terminates at the Receiver (100b or a Reserved value), or if the Message Code field value is defined and corresponds to a Message which must be comprehended by the Switch, the Switch must process the Message according to the Message processing rules
- If the value in r[2:0] indicates Gathered and routed to Root Complex (101b), see Section 5.3.3.2.1 for Message handling rules
- It is an error to receive any Msg/MsgD Request other than a PME_TO_Ack that specifies 101b routing. It is an error to receive a PME_TO_Ack at the Upstream Port of a Switch. Switches may optionally check for violations of these rules. These checks are independently optional (see Section 6.2.3.4). If checked, violations are Malformed TLPs, and are reported errors associated with the Receiving Port (see Section 6.2).

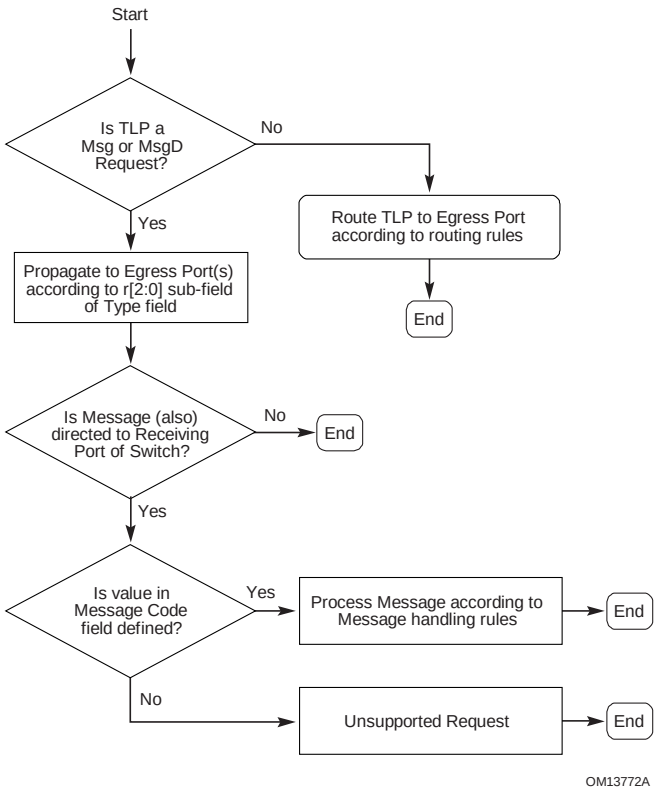


Figure 2-38: Flowchart for Switch Handling of TLPs

2.3.1. Request Handling Rules

This section describes how Received Requests are handled, following the initial processing done with all TLPs. The rules are diagramed in the flowchart shown in

Figure 2-39.

- If the Request Type is not supported (by design or because of configuration settings) by the device, the Request is an Unsupported Request, and is reported according to Section 6.2
 - If the Request requires Completion, a Completion Status of UR is returned (see Section 2.2.8.10)

IMPLEMENTATION NOTE

When Requests are Terminated Using Unsupported Request

In Conventional PCI, a device "claims" a request on the bus by asserting DEVSEL#. If no device claims a request after a set number of clocks, the request is terminated as a Master Abort. Since PCI Express is a point to point interconnect, there is no equivalent mechanism for claiming a request on a Link, since all transmissions by one component are always sent to the other component on the Link. Therefore, it is necessary for the receiver of a request to determine if the request should be "claimed." If the request is not claimed, then it is handled as an Unsupported Request, which is the PCI Express equivalent of conventional PCI's Master Abort termination. In general, one can determine the correct behavior by asking the question: *Would the device assert DEVSEL# for this request in conventional PCI?*

For device Functions with Type 0 headers (all types of Endpoints), it is relatively simple to answer this question. For Memory and I/O Requests, this determination is based on the address ranges the Function has been programmed to respond to. For Configuration requests, the Type 0 request format indicates the device is by definition the "target", although the device will still not claim the Configuration Request if it addresses an unimplemented Function.

For device Functions with Type 1 headers (Root Ports, Switches and Bridges), the same question can generally be applied, but since the behavior of a conventional PCI bridge is more complicated than that of a Type 0 Function, it is somewhat more difficult to determine the answers. One must consider Root Ports and Switch Ports as if they were actually composed of conventional PCI to PCI bridges, and then at each stage consider the configuration settings of the virtual bridge to determine the correct behavior.

PCI Express Messages do not exist in conventional PCI, so the above guideline cannot be applied. This specification describes specifically for each type of Message when a device must handle the request as an Unsupported Request. Messages pass through Root and Switch Ports unaffected by conventional PCI control mechanisms including Bus Master Enable and power state setting.

Note that Completer Abort, which is the PCI Express equivalent to Target Abort, is used only to indicate a serious error that makes the completer permanently unable to respond to a request that it would otherwise have normally responded to. Since Target Abort is used in conventional PCI only when a target has asserted DEVSEL#, is incorrect to use Completer Abort for any case where a conventional PCI target would have ignored a request by not asserting DEVSEL#.

- If the Request is a Message, and the Message Code, routing field, or Msg / MsgD indication corresponds to a combination ~~specifies a value~~ that is undefined, or that corresponds to a Message not supported by the device Function, (other than Vendor_Defined Type 1 which is not treated as an error – see Section 2.2.8.6), the Request is an Unsupported Request, and is reported according to Section 6.2
 - If the Message Code is a supported value, process the Message according to the corresponding Message processing rules; if the Message Code is an Ignored Message and the receiver is ignoring it, ignore the Message without reporting any error (see Section 2.2.8.7)

Comment [VS75]: Errata A11

If the Request is not a Message, and is a supported Type, specific implementations may be optimized based on a defined programming model which ensures that certain types of (otherwise legal) Requests will never occur. Such implementations may take advantage of the following rule:

- If the Request violates the programming model of the device Function, the Function may optionally treat the Request as a Completer Abort, instead of handling the Request normally
 - If the Request is treated as a Completer Abort, this is a reported error associated with the Function (see Section 6.2)
 - If the Request requires Completion, a Completion Status of CA is returned (see Section 2.2.8.10)



IMPLEMENTATION NOTE

Optimizations Based on Restricted Programming Model

When a device's programming model restricts (vs. what is otherwise permitted in PCI Express) the characteristics of a Request, that device is permitted to "Completer Abort" any Requests which violate the programming model. Examples include unaligned or wrong-size access to a register block and unsupported size of request to a Memory Space.

Generally, devices are able to assume a restricted programming model when all communication will be between the device's driver software and the device itself. Devices which may be accessed directly by operating system software or by applications which may not comprehend the restricted programming model of the device (typically devices which implement legacy capabilities) should be designed to support all types of Requests which are possible in the existing usage model for the device. If this is not done, the device may fail to operate with existing software.

If the Request arrives between the time an FLR has been initiated and the completion of the FLR by the targeted Function, the Request is permitted to be silently discarded (following update of flow control credits) without logging or signaling it as an error. It is recommended that the Request be handled as an Unsupported Request (UR).

- Otherwise (supported Request Type, not a Message), process the Request
 - If the Completer is permanently unable to process the Request due to a device-specific error condition the Completer must, if possible, handle the Request as a Completer Abort
 - This is a reported error associated with the Receiving Function, if the error can be isolated to a specific Function in the component, or to the Receiving Port if the error cannot be isolated (see Section 6.2)
 - For Configuration Requests only, following reset it is possible for a device to terminate the request but indicate that it is temporarily unable to process the Request, but will be able to process the Request in the future – in this case, the Configuration Request Retry Status (CRS) Completion Status is used (see Section 6.6). Valid reset conditions after which a device is permitted to return CRS are:
 - Cold, Warm, and Hot Resets
 - FLRs
 - A reset initiated in response to a $D3_{hot}$ to $D0_{uninitialized}$ device state transition.
 - A device Function is explicitly not permitted to return CRS following a software-initiated reset (other than an FLR) of the device, e.g., by the device's software driver writing to a device-specific reset bit. A device Function is not permitted to return CRS after it has indicated that it is Configuration-Ready (see Section 6.23.) without an intervening valid reset (i.e., FLR or Conventional Reset) condition, or if the Immediate Readiness bit in the Function's Status register is Set. Additionally, a device Function is not permitted to return CRS after having previously returned a Successful Completion without an intervening valid reset (i.e., FLR or Conventional Reset) condition.
 - In the process of servicing the Request, the Completer may determine that the (otherwise acceptable) Request must be handled as an error, in which case the Request is handled according to the type of the error
 - Example: A PCI Express/PCI Bridge may initially accept a Request because it specifies a Memory Space range mapped to the secondary side of the Bridge, but the Request may Master Abort or Target Abort on the PCI side of the Bridge. From the PCI Express perspective, the status of the Request in this case is UR (for Master Abort) or CA (for Target Abort). If the Request requires Completion on PCI Express, the corresponding Completion Status is returned.
- If the Request is a type which requires a Completion to be returned, generate a Completion according to the rules for Completion Formation (see Section 2.2.9)
 - The Completion Status is determined by the result of handling the Request
 - If the Request has an ECRC Check Failed error, then it is implementation-specific whether to return a Completion or not, and if so, which of the architected values to use for its Completion Status. However, it is strongly recommended that the Completer return a Completion with a UR Completion Status.

Comment [VS76]: Para updated per RC_ECN_RN_27_Aug_2013

Comment [VS77]: Errata A24 (Part C list resolved 10/13)

- Under normal operating conditions, PCI Express Endpoints and Legacy Endpoints must never delay the acceptance of a Posted Request for more than 10 µs, which is called the Posted Request Acceptance Limit. The device must either (a) be designed to process received Posted Requests and return associated Flow Control credits within the necessary time limit, or (b) rely on a restricted programming model to ensure that a Posted Request is never sent to the device either by software or by other devices while the device is unable to accept a new Posted Request within the necessary time limit.
 - The following are not considered normal operating conditions under which the Posted Request Acceptance Limit applies:
 - The period immediately following a Fundamental Reset (see Section 6.6).
 - TLP retransmissions or Link retraining.
 - One or more dropped FCPs.
 - The device being in a diagnostic mode.
 - The device being in a device-specific mode that is not intended for normal use.
 - The following are considered normal operating conditions, but any delays they cause do not count against the Posted Request Acceptance Limit:
 - Upstream TLP traffic delaying Upstream FCPs.
 - The Link coming out of a low-power state.
 - Arbitration with traffic on other VCs.
 - Though not a requirement, it is strongly recommended that Root Complex Integrated Endpoints also honor the Posted Request Acceptance Limit.
 - If the device supports being a target for I/O writes, which are Non-Posted Requests, it is strongly recommended that each associated Completion be returned within the same time limit as for Posted Request acceptance, although this is not a requirement.



IMPLEMENTATION NOTE

Restricted Programming Model for Meeting the Posted Request Acceptance Limit

Some hardware designs may not be able to process every Posted Request within the required acceptance time limit. An example is writing to a command queue where commands can take longer than the acceptance time limit to complete. Subsequent writes to such a device when it is currently processing a previous write could experience acceptance delays that exceed the limit. Such devices may rely on a restricted programming model, where the device driver limits the rate of memory writes to the device, the driver polls the device to determine buffer availability before issuing the write transaction, or the driver implements some other software-based flow control mechanism.

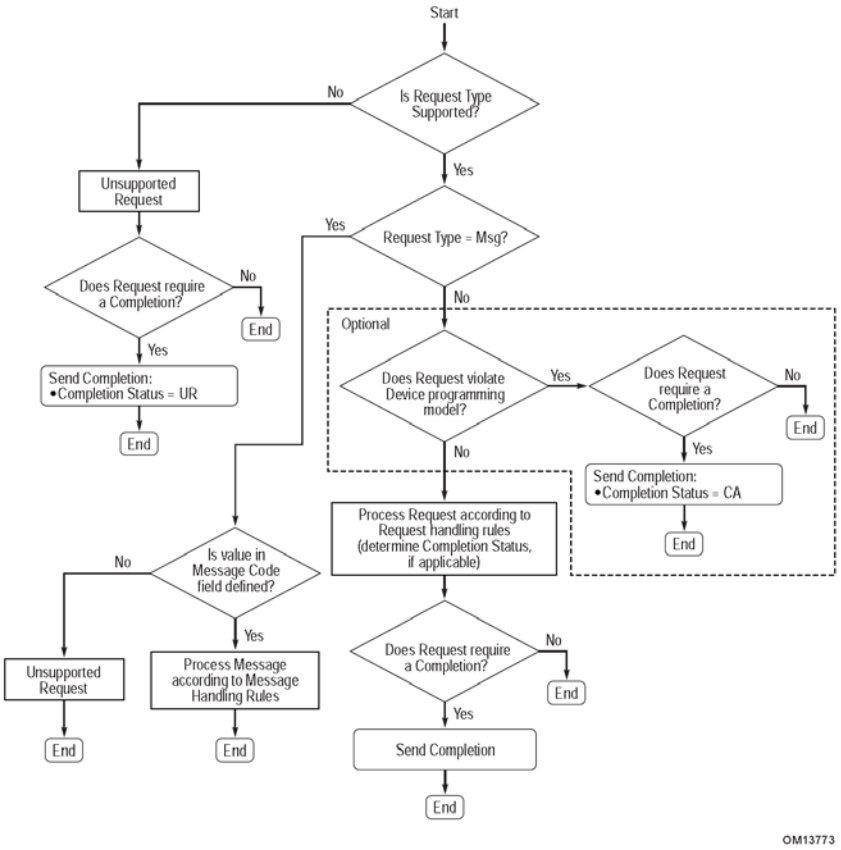


Figure 2-39: Flowchart for Handling of Received Request

IMPLEMENTATION NOTE

Configuration Request Retry Status

Some devices require a lengthy self-initialization sequence to complete before they are able to service Configuration Requests (common with intelligent I/O solutions on PCI). PCI/PCI-X architecture has specified a 2^{25} (PCI) or 2^{26} (PCI-X) clock "recovery time" T_{rhfa} following reset to provide the required self-initialization time for such devices. PCI Express "softens" the need for this time based recovery period by implementing a Configuration Request Retry Status (CRS) Completion Status. A device in receipt of a Configuration Request following a valid reset condition may respond with a CRS Completion Status to terminate the Request, and thus effectively stall the Configuration Request until such time that the subsystem has completed local initialization and is ready to communicate with the host. Note that it is only legal to respond with a CRS Completion Status in response to a Configuration Request. Sending this Completion Status in response to any other Request type is illegal (see Section 2.3.2). Readiness Notifications (see Section 6.23.) and Immediate Readiness (see Conventional PCI Specification Revision 3.0 Section 6.2.3. and PCI-PM Specification Revision 1.2 Section 3.2.3) also forbid the use of CRS Completion Status in certain situations.

Comment [VS78]: D-ECN-Readiness Notifications

Receipt by the Requester of a Completion with CRS Completion Status terminates the Configuration Request on PCI Express. Further action by the Root Complex regarding the original Configuration Request is specified in Section 2.3.2.

Root Complexes that implement CRS Software Visibility have the ability to report the receipt of CRS Completion Status to software, enabling software to attend to other tasks rather than being stalled while the device completes its self-initialization. Software that intends to take advantage of this mechanism must ensure that the first access made to a device following a valid reset condition is a Configuration Read Request accessing both bytes of the Vendor ID field in the device's Configuration Space header. For this case only, the Root Complex, if enabled, will synthesize a special read-data value for the Vendor ID field to indicate to software that CRS Completion Status has been returned by the device. For other Configuration Requests, or when CRS Software Visibility is not enabled, the Root Complex will generally re-issue the Configuration Request until it completes with a status other than CRS as described in Section 2.3.2.

When used in systems including PCI Express to PCI/PCI-X Bridges, system software and/or the Root Complex must comprehend the limit T_{rhfa} for PCI/PCI-X agents as described in Sections 2.8 and 6.6. Similarly, systems using PCI Express components which require additional self initialization time beyond the minimum guaranteed must provide some mechanism for re-issuing Configuration Requests terminated with CRS status. In systems running legacy PCI/PCI-X based software, the Root Complex must re-issue the Configuration Request using a hardware mechanism to ensure proper enumeration of the system.

Refer to Section 6.6 for more information on reset.

2.3.1.1. Data Return for Read Requests

- Individual Completions for Memory Read Requests may provide less than the full amount of data Requested so long as all Completions for a given Request when combined return exactly the amount of data Requested in the Read Request.
 - Completions for different Requests cannot be combined.
 - I/O and Configuration Reads must be completed with exactly one Completion.
 - The Completion Status for a Completion corresponds only to the status associated with the data returned with that Completion
 - A Completion with status other than Successful Completion terminates the Completions for a single Read Request
 - In this case, the value in the Length field is undefined, and must be ignored by the Receiver
- Completions must not include more data than permitted by Max_Payload_Size.
 - Receivers must check for violations of this rule. Refer to Section 2.2.

 Note: This is simply a specific case of the rules which apply to all TLPs with data payloads
- Memory Read Requests may be completed with one, or in some cases, multiple Completions
- Read Completion Boundary (RCB) determines the naturally aligned address boundaries on which a Read Request may be serviced with multiple Completions
 - For a Root Complex, RCB is 64 bytes or 128 bytes
 - This value is reported in the Link Control register (see Section 7.8.7)

 Note: Bridges and Endpoints may implement a corresponding command bit which may be set by system software to indicate the RCB value for the Root Complex, allowing the Bridge or Endpoint to optimize its behavior when the Root Complex's RCB is 128 bytes.
 - For all other System Elements, RCB is 128 bytes
- Completions for Requests which do not cross the naturally aligned address boundaries at integer multiples of RCB bytes must include all data specified in the Request

- Requests which do cross the address boundaries at integer multiples of RCB bytes may be completed using more than one Completion, but the data must not be fragmented except along the following address boundaries:
 - The first Completion must start with the address specified in the Request, and must end at one of the following:
 - the address specified in the Request plus the length specified by the Request (i.e., the entire Request)
 - an address boundary between the start and end of the Request at an integer multiple of RCB bytes
 - The final Completion must end with the address specified in the Request plus the length specified by the Request
 - All Completions between, but not including, the first and final Completions must be an integer multiple of RCB bytes in length
- Receivers may optionally check for violations of RCB. If a Receiver implementing this check determines that a Completion violates this rule, it must handle the Completion as a Malformed TLP
 - This is a reported error associated with the Receiving Port (see Section 6.2)
- Multiple Memory Read Completions for a single Read Request must return data in increasing address order.
- For each Memory Read Completion, the Byte Count field must indicate the remaining number of bytes required to complete the Request including the number of bytes returned with the Completion, except when the BCM bit is 1b.[21]
 - The total number of bytes required to complete a Memory Read Request is calculated as shown in Table 2-37
 - If a Memory Read Request is completed using multiple Completions, the Byte Count value for each successive Completion is the value indicated by the preceding Completion minus the number of bytes returned with the preceding Completion.

[21] Only PCI-X completers set the BCM bit to 1b.



IMPLEMENTATION NOTE

BCM Bit Usage

To satisfy certain PCI-X protocol constraints, a PCI-X Bridge or PCI-X Completer for a PCI-X burst read in some cases will set the Byte Count field in the first PCI-X transaction of the Split Completion sequence to indicate the size of just that first transaction instead of the entire burst read. When this occurs, the PCI-X Bridge/PCI-X Completer will also set the BCM bit in that first PCI-X transaction, to indicate that the Byte Count field has been modified from its normal usage. Refer to the *PCI-X Addendum to the PCI Local Bus Specification, Revision 2.0* for further details.

A PCI Express Memory Read Requester needs to correctly handle the case when a PCI-X Bridge/PCI-X Completer sets the BCM bit. When this occurs, the first Read Completion packet returned to the Requester will have the BCM bit set, indicating that the Byte Count field reports the size of just that first packet instead of the entire remaining byte count. The Requester should not conclude at this point that other packets of the Read Completion are missing.

The BCM bit will never be set in subsequent packets of the Read Completion, so the Byte Count field in those subsequent packets will always indicate the remaining byte count in each instance. Thus, the Requester can use the Byte Count field in these packets to determine if other packets of the Read Completion are missing.

PCI Express Completers will never set the BCM bit.

Table 2-37: Calculating Byte Count from Length and Byte Enables

1st DW BE[3:0] (b)	Last DW BE[3:0] (b)	Total Byte Count
1xx1	0000[22]	4
01x1	0000	3
1x10	0000	3
0011	0000	2
0110	0000	2
1100	0000	2
0001	0000	1
0010	0000	1
0100	0000	1
1000	0000	1
0000	0000	1
xxx1	1xxx	Length[23] * 4
xxx1	01xx	(Length * 4) - 1
xxx1	001x	(Length * 4) - 2
xxx1	0001	(Length * 4) - 3
xx10	1xxx	(Length * 4) - 1
xx10	01xx	(Length * 4) - 2
xx10	001x	(Length * 4) - 3
xx10	0001	(Length * 4) - 4
x100	1xxx	(Length * 4) - 2
x100	01xx	(Length * 4) - 3
x100	001x	(Length * 4) - 4
x100	0001	(Length * 4) - 5
1000	1xxx	(Length * 4) - 3
1000	01xx	(Length * 4) - 4
1000	001x	(Length * 4) - 5
1000	0001	(Length * 4) - 6

[22] Note that Last DW BE of 0000b is permitted only with a Length of 1 DW.

[23] Length is the number of DW as indicated by the value in the Length field, and is multiplied by 4 to yield a number in bytes.

- For all Memory Read Completions, the Lower Address field must indicate the lower bits of the byte address for the first enabled byte of data returned with the Completion
 - For the first (or only) Completion, the Completer can generate this field from the least significant 5 bits of the address of the Request concatenated with 2 bits of byte-level address formed as shown in Table 2-38.
 - For any subsequent Completions, the Lower Address field will always be zero except for Completions generated by a Root Complex with an RCB value of 64 bytes. In this case the least significant 6 bits of the Lower Address field will always be zero and the most significant bit of the Lower Address field will toggle according to the alignment of the 64-byte data payload.

Table 2-38: Calculating Lower Address from 1st DW BE

1st DW BE[3:0] (b)	Lower Address[1:0] (b)
0000	00
xxx1	00
xx10	01
x100	10
1000	11

- When a Read Completion is generated with a Completion Status other than Successful Completion:
 - No data is included with the Completion
 - The Cpl (or CplLk) encoding is used instead of CplD (or CplDLk)
 - This Completion is the final Completion for the Request.
 - The Completer must not transmit additional Completions for this Request.
 - Example: Completer split the Request into four parts for servicing; the second Completion had a Completer Abort Completion Status; the Completer terminated servicing for the Request, and did not Transmit the remaining two Completions.
 - The Byte Count field must indicate the remaining number of bytes that would be required to complete the Request (as if the Completion Status were Successful Completion)
 - The Lower Address field must indicate the lower bits of the byte address for the first enabled byte of data that would have been returned with the Completion if the Completion Status were Successful Completion



IMPLEMENTATION NOTE

Restricted Programming Model

When a device's programming model restricts (vs. what is otherwise permitted in PCI Express) the size and/or alignment of Read Requests directed to the device, that device is permitted to use a Completer Abort Completion Status for Read Requests which violate the programming model. An implication of this is that such devices, generally devices where all communication will be between the device's driver software and the device itself, need not necessarily implement the buffering required to generate Completions of length RCB. However, in all cases, the boundaries specified by RCB must be respected for all reads which the device will Complete with Successful Completion status.

Examples:

1. Memory Read Request with Address of 1 0000h and Length of C0h bytes (192 decimal) could be completed by a Root Complex with an RCB value of 64 bytes with one of the following combinations of Completions (bytes):
192 –or– 128, 64 –or– 64, 128 –or– 64, 64, 64

2. Memory Read Request with Address of 1 0000h and Length of C0h bytes (192 decimal) could be completed by a Root Complex with an RCB value of 128 bytes in one of the following combinations of Completions (bytes):
192 –or– 128, 64

3. Memory Read Request with Address of 1 0020h and Length of 100h bytes (256 decimal) could be completed by a Root Complex with an RCB value of 64 bytes in one of the following combinations of Completions (bytes):
256 –or–
32, 224 –or– 32, 64, 160 –or– 32, 64, 64, 96 –or– 32, 64, 64, 64, 32 –or–
32, 64, 128, 32 –or– 32, 128, 96 –or– 32, 128, 64, 32 –or–
96, 160 –or– 96, 128, 32 –or– 96, 64, 96 –or– 96, 64, 64, 32 –or–
160, 96 –or– 160, 64, 32 –or– 224, 32

4. Memory Read Request with Address of 1 0020h and Length of 100h bytes (256 decimal) could be completed by an Endpoint in one of the following combinations of Completions (bytes):
256 –or– 96, 160 –or– 96, 128, 32 –or– 224, 32

2.3.2. Completion Handling Rules

- When a device receives a Completion which does not match the Transaction ID for any of the outstanding Requests issued by that device, the Completion is called an "Unexpected Completion."
- If a received Completion matches the Transaction ID of an outstanding Request, but in some other way does not match the corresponding Request (e.g., a problem with Attributes, Traffic Class, Byte Count, Lower Address, etc), it is strongly recommended for the Receiver to handle the Completion as a Malformed TLP. However, if the Completion is otherwise properly

formed, it is permitted[24] for the Receiver to handle the Completion as an Unexpected Completion.

- When an Ingress Port of a Switch receives a Completion which cannot be forwarded, that Ingress Port must handle the Completion as an Unexpected Completion. This includes Completions that target:
 - a non-existent Function in the Device associated with the Upstream Port,
 - a non-existent Device on the Bus associated with the Upstream Port,
 - a non-existent Device or Function on the internal switching fabric, or
 - a Bus Number within the Upstream Port's Bus Number aperture but not claimed by any Downstream Port.
- Receipt of an Unexpected Completion is an error and must be handled according to the following rules:
 - The agent receiving an Unexpected Completion must discard the Completion.
 - An Unexpected Completion is a reported error associated with the Receiving Port (see Section 6.2).

 Note: Unexpected Completions are assumed to occur mainly due to Switch misrouting of the Completion. The Requester of the Request may not receive a Completion for its Request in this case, and the Requester's Completion Timeout mechanism (see Section 2.8) will terminate the Request.
- Completions with a Completion Status other than Successful Completion or Configuration Request Retry Status (in response to Configuration Request only) must cause the Requester to:
 - Free Completion buffer space and other resources associated with the Request.
 - Handle the error via a Requester-specific mechanism (see Section 6.2.3.2.5).

 If the Completion arrives between the time an FLR has been initiated and the completion of the FLR by the targeted Function, the Completion is permitted to be handled as an Unexpected Completion or to be silently discarded (following update of flow control credits) without logging or signaling it as an error. Once the FLR has completed, received Completions corresponding to Requests issued prior to the FLR must be handled as Unexpected Completions, unless the Function has been re-enabled to issue Requests.
- Root Complex handling of a Completion with Configuration Request Retry Status for a Configuration Request is implementation specific, except for the period following system reset (see Section 6.6). For Root Complexes that support CRS Software Visibility, the following rules apply:
 - If CRS Software Visibility is not enabled, the Root Complex must re-issue the Configuration Request as a new Request.
 - If CRS Software Visibility is enabled (see below):

[24] For the case where only the Byte Count or Lower Address fields mismatch the expected values for a Memory Read Request, it is actually recommended for the Receiver to handle the Completion as an Unexpected Completion, since the mismatch might be caused by a previous Completion being misrouted.

- For a Configuration Read Request that includes both bytes of the Vendor ID field of a device Function's Configuration Space Header, the Root Complex must complete the Request to the host by returning a read-data value of 0001h for the Vendor ID field and all '1's for any additional bytes included in the request. This read-data value has been reserved specifically for this use by the PCI-SIG and does not correspond to any assigned Vendor ID.
- For a Configuration Write Request or for any other Configuration Read Request, the Root Complex must re-issue the Configuration Request as a new Request.

A Root Complex implementation may choose to limit the number of Configuration Request/CRS Completion Status loops before determining that something is wrong with the target of the Request and taking appropriate action, e.g., complete the Request to the host as a failed transaction.

CRS Software Visibility may be enabled through the CRS Software Visibility Enable bit in the Root Control register (see Section 7.8.12) to control Root Complex behavior on an individual Root Port basis. Alternatively, Root Complex behavior may be managed through the CRS Software Visibility Enable bit in the RCRB Control register as described in Section 7.20.4, permitting the behavior of one or more Root Ports or Root Complex Integrated Endpoints to be controlled by a single Enable bit. For this alternate case, each Root Port or Root Complex Integrated Endpoint declares its association with a particular Enable bit via an RCRB header association in a Root Complex Link Declaration Capability (see Section 7.13). Each Root Port or Root Complex Integrated Endpoint is permitted to be controlled by at most one Enable bit. Thus, for example, it is prohibited for a Root Port whose Root Control register contains an Enable bit to declare an RCRB header association to an RCRB that also includes an Enable bit in its RCRB Header Capability. The presence of an Enable bit in a Root Port or RCRB Header Capability is indicated by the corresponding CRS Software Visibility bit (see Sections 7.8.13 and 7.20.3, respectively).

- Completions with a Configuration Request Retry Status in response to a Request other than a Configuration Request are illegal. Receivers may optionally report these violations as Malformed TLPs
 - This is a reported error associated with the Receiving Port (see Section 6.2)
- Completions with a Reserved Completion Status value are treated as if the Completion Status was Unsupported Request (UR)
- Completions with a Completion Status of Unsupported Request or Completer Abort are reported using the conventional PCI reporting mechanisms (see Section 7.5.1.2)
 - Note that the error condition that triggered the generation of such a Completion is reported by the Completer as described in Section 6.2
- When a Read Completion or an AtomicOp Completion is received with a Completion Status other than Successful Completion:
 - No data is included with the Completion
 - The Cpl (or CplLk) encoding is used instead of CplD (CplDLk)
 - This Completion is the final Completion for the Request.

- The Requester must consider the Request terminated, and not expect additional Completions.
 - Handling of partial Completions Received earlier is implementation specific.

 Example: The Requester received 32 bytes of Read data for a 128-byte Read Request it had issued, then a Completion with the Completer Abort Completion Status. The Requester then must free the internal resources which had been allocated for that particular Read Request.



IMPLEMENTATION NOTE

Read Data Values with UR Completion Status

Some system configuration software depends on reading a data value of all 1's when a Configuration Read Request is terminated as an Unsupported Request, particularly when probing to determine the existence of a device in the system. A Root Complex intended for use with software that depends on a read-data value of all 1's must synthesize this value when UR Completion Status is returned for a Configuration Read Request.

2.4. Transaction Ordering

2.4.1. Transaction Ordering Rules

Table 2-39 defines the ordering requirements for PCI Express Transactions. The rules defined in this table apply uniformly to all types of Transactions on PCI Express including Memory, I/O, Configuration, and Messages. The ordering rules defined in this table apply within a single Traffic Class (TC). There is no ordering requirement among transactions with different TC labels. Note that this also implies that there is no ordering required between traffic that flows through different Virtual Channels since transactions with the same TC label are not allowed to be mapped to multiple VCs on any PCI Express Link.

For Table 2-39, the columns represent a first issued transaction and the rows represent a subsequently issued transaction. The table entry indicates the ordering relationship between the two transactions. The table entries are defined as follows:

- Yes–the second transaction (row) must be allowed to pass the first (column) to avoid deadlock. (When blocking occurs, the second transaction is required to pass the first transaction. Fairness must be comprehended to prevent starvation.)
- Y/N–there are no requirements. The second transaction may optionally pass the first transaction or be blocked by it.
- No–the second transaction must not be allowed to pass the first transaction. This is required to support the Producer-Consumer strong ordering model.

Table 2-39: Ordering Rules Summary

Row Pass Column?		Posted Request (Col 2)	Non-Posted Request		Completion (Col 5)
			Read Request (Col 3)	NPR with Data (Col 4)	
Posted Request (Row A)		a) No b) Y/N	Yes	Yes	a) Y/N b) Yes
Non-Posted Request	**Read Request** (Row B)	a) No b) Y/N	Y/N	Y/N	Y/N
	NPR with Data (Row C)	a) No b) Y/N	Y/N	Y/N	Y/N
Completion (Row D)		a) No b) Y/N	Yes	Yes	a) Y/N b) No

Explanation of the row and column headers in Table 2-39:

A **Posted Request** is a Memory Write Request or a Message Request.

A **Read Request** is a Configuration Read Request, an I/O Read Request, or a Memory Read Request.

An **NPR** (Non-Posted Request) **with Data** is a Configuration Write Request, an I/O Write Request, or an AtomicOp Request.

A **Non-Posted Request** is a Read Request or an NPR with Data.

Explanation of the entries in Table 2-39:

A2a	A Posted Request must not pass another Posted Request unless A2b applies.
A2b	A Posted Request with RO[25] Set is permitted to pass another Posted Request.[26] A Posted Request with IDO Set is permitted to pass another Posted Request if the two Requester IDs are different or if both Requests contain a PASID TLP Prefix and the two PASID values are different..
A3, A4	A Posted Request must be able to pass Non-Posted Requests to avoid deadlocks.
A5a	A Posted Request is permitted to pass a Completion, but is not required to be able to pass Completions unless A5b applies.
A5b	Inside a PCI Express to PCI/PCI-X Bridge whose PCI/PCI-X bus segment is operating in conventional PCI mode, for transactions traveling in the PCI Express to PCI direction, a Posted Request must be able to pass Completions to avoid deadlock.
B2a	A Read Request must not pass a Posted Request unless B2b applies.
B2b	A Read Request with IDO Set is permitted to pass a Posted Request if the two Requester IDs are different or if both Requests contain a PASID TLP Prefix and the two PASID values are different..
C2a	An NPR with Data must not pass a Posted Request unless C2b applies.
C2b	An NPR with Data and with RO Set[27] is permitted to pass Posted Requests. An NPR with Data and with IDO Set is permitted to pass a Posted Request if the two Requester IDs are different or if both Requests contain a PASID TLP Prefix and the two PASID values are different.
B3, B4, C3, C4	A Non-Posted Request is permitted to pass another Non-Posted Request.
B5, C5	A Non-Posted Request is permitted to pass a Completion.
D2a	A Completion must not pass a Posted Request unless D2b applies.
D2b	An I/O or Configuration Write Completion[28] is permitted to pass a Posted Request. A Completion with RO Set is permitted to pass a Posted Request. A Completion with

Comment [VS79]: ECN: PASID

Comment [VS80]: ECN: PASID

Comment [VS81]: ECN: PASID

[25] In this section, "RO" is an abbreviation for the Relaxed Ordering Attribute field.

[26] Some usages are enabled by not implementing this passing (see the No RO-enabled PR-PR Passing bit in Section 7.8.15).

[27] Note: Not all NPR with Data transactions are permitted to have RO Set.

[28] Note: Not all components can distinguish I/O and Configuration Write Completions from other Completions. In particular, routing elements not serving as the associated Requester or Completer generally cannot make this distinction. A component must not apply this rule for I/O and Configuration Write Completions unless it is certain of the associated Request type.

	IDO Set is permitted to pass a Posted Request if the Completer ID of the Completion is different from the Requester ID of the Posted Request.
D3, D4	A Completion must be able to pass Non-Posted Requests to avoid deadlocks.
D5a	Completions with different Transaction IDs are permitted to pass each other.
D5b	Completions with the same Transaction ID must not pass each other. This ensures that multiple Completions associated with a single Memory Read Request will remain in ascending address order.

Additional Rules:

- PCI Express Switches are permitted to allow a Memory Write or Message Request with the Relaxed Ordering bit set to pass any previously posted Memory Write or Message Request moving in the same direction. Switches must forward the Relaxed Ordering attribute unmodified. The Root Complex is also permitted to allow data bytes within the Request to be written to system memory in any order. (The bytes must be written to the correct system memory locations. Only the order in which they are written is unspecified).
- For Root Complex and Switch, Memory Write combining (as defined in the *PCI Local Bus Specification*) is prohibited.
- Note: This is required so that devices can be permitted to optimize their receive buffer and control logic for Memory Write sizes matching their natural expected sizes, rather than being required to support the maximum possible Memory Write payload size.
- Combining of Memory Read Requests, and/or Completions for different Requests is prohibited.
- The No Snoop bit does not affect the required ordering behavior.
- For Root Ports and Switch Downstream Ports, acceptance of a Posted Request or Completion must not depend upon the transmission of a Non-Posted Request within the same traffic class.[29]
- For Switch Upstream Ports, acceptance of a Posted Request or Completion must not depend upon the transmission on a Downstream Port of Non-Posted Request within the same traffic class.[29]
- For Endpoint, Bridge, and Switch Upstream Ports, the acceptance of a Posted Request must not depend upon the transmission of any TLP from that same Upstream Port within the same traffic class.[29]
- For Endpoint, Bridge, and Switch Upstream Ports, the acceptance of a Non-posted Request must not depend upon the transmission of a Non-Posted Request from that same Upstream Port within the same traffic class.[29]

[29] Satisfying the above rules is a necessary, but not sufficient condition to ensure deadlock free operation. Deadlock free operation is dependent upon the system topology, the number of Virtual Channels supported and the configured Traffic Class to Virtual Channel mappings. Specification of platform and system constraints to ensure deadlock free operation is outside the scope of this specification (see Appendix D for a discussion of relevant issues).

- For Endpoint, Bridge, and Switch Upstream Ports, the acceptance of a Completion must not depend upon the transmission of any TLP from that same Upstream Port within the same traffic class.[29]

Note that Endpoints are never permitted to block acceptance of a Completion

- Completions issued for Non-Posted requests must be returned in the same Traffic Class as the corresponding Non-Posted request.
- Root Complexes that support peer-to-peer operation and Switches must enforce these transaction ordering rules for all forwarded traffic.

To ensure deadlock-free operation, devices should not forward traffic from one Virtual Channel to another. The specification of constraints used to avoid deadlock in systems where devices forward or translate transactions between Virtual Channels is outside the scope of this document (see Appendix D for a discussion of relevant issues).



IMPLEMENTATION NOTE

Large Memory Reads vs. Multiple Smaller Memory Reads

Note that the rule associated with entry D5b in Table 2-39 ensures that for a single Memory Read Request serviced with multiple Completions, the Completions will be returned in address order. However, the rule associated with entry D5a permits that different Completions associated with distinct Memory Read Requests may be returned in a different order than the issue order for the Requests. For example, if a device issues a single Memory Read Request for 256 bytes from location 1000h, and the Request is returned using two Completions (see Section 2.3.1.1) of 128 bytes each, it is guaranteed that the two Completions will return in the following order:

1st Completion returned: Data from 1000h to 107Fh.

2nd Completion returned: Data from 1080h to 10FFh.

However, if the device issues two Memory Read Requests for 128 bytes each, first to location 1000h, then to location 1080h, the two Completions may return in either order:

1st Completion returned: Data from 1000h to 107Fh.

2nd Completion returned: Data from 1080h to 10FFh.

– or –

1st Completion returned: Data from 1080h to 10FFh.

2nd Completion returned: Data from 1000h to 107Fh.

2.4.2. Update Ordering and Granularity Observed by a Read Transaction

If a Requester using a single transaction reads a block of data from a Completer, and the Completer's data buffer is concurrently being updated, the ordering of multiple updates and granularity of each update reflected in the data returned by the read is outside the scope of this

specification. This applies both to updates performed by PCI Express write transactions and updates performed by other mechanisms such as host CPUs updating host memory.

If a Requester using a single transaction reads a block of data from a Completer, and the Completer's data buffer is concurrently being updated by one or more entities not on the PCI Express fabric, the ordering of multiple updates and granularity of each update reflected in the data returned by the read is outside the scope of this specification.

As an example of update ordering, assume that the block of data is in host memory, and a host CPU writes first to location A and then to a different location B. A Requester reading that data block with a single read transaction is not guaranteed to observe those updates in order. In other words, the Requester may observe an updated value in location B and an old value in location A, regardless of the placement of locations A and B within the data block. Unless a Completer makes its own guarantees (outside this specification) with respect to update ordering, a Requester that relies on update ordering must observe the update to location B via one read transaction before initiating a subsequent read to location A to return its updated value.

As an example of update granularity, if a host CPU writes a QWORD to host memory, a Requester reading that QWORD from host memory may observe a portion of the QWORD updated and another portion of it containing the old value.

While not required by this specification, it is strongly recommended that host platforms guarantee that when a host CPU writes aligned DWORDs or aligned QWORDs to host memory, the update granularity observed by a PCI Express read will not be smaller than a DWORD.



IMPLEMENTATION NOTE

No Ordering Required Between Cachelines

A Root Complex serving as a Completer to a single Memory Read that requests multiple cachelines from host memory is permitted to fetch multiple cachelines concurrently, to help facilitate multi-cacheline completions, subject to Max_Payload_Size. No ordering relationship between these cacheline fetches is required.

2.4.3. Update Ordering and Granularity Provided by a Write Transaction

If a single write transaction containing multiple DWORDs and the Relaxed Ordering bit clear is accepted by a Completer, the observed ordering of the updates to locations within the Completer's data buffer must be in increasing address order. This semantic is required in case a PCI or PCI-X Bridge along the path combines multiple write transactions into the single one. However, the observed granularity of the updates to the Completer's data buffer is outside the scope of this specification.

While not required by this specification, it is strongly recommended that host platforms guarantee that when a PCI Express write updates host memory, the update granularity observed by a host CPU will not be smaller than a DWORD.

As an example of update ordering and granularity, if a Requester writes a QWORD to host memory, in some cases a host CPU reading that QWORD from host memory could observe the first DWORD updated and the second DWORD containing the old value.

2.5. Virtual Channel (VC) Mechanism

The Virtual Channel (VC) mechanism provides support for carrying, throughout the fabric, traffic that is differentiated using TC labels. The foundation of VCs are independent fabric resources (queues/buffers and associated control logic). These resources are used to move information across Links with fully independent flow-control between different VCs. This is key to solving the problem of flow-control induced blocking where a single traffic flow may create a bottleneck for all traffic within the system.

Traffic is associated with VCs by mapping packets with particular TC labels to their corresponding VCs. The VC and MFVC mechanisms allow flexible mapping of TCs onto the VCs. In the simplest form, TCs can be mapped to VCs on a 1:1 basis. To allow performance/cost tradeoffs, PCI Express provides the capability of mapping multiple TCs onto a single VC. Section 2.5.2 covers details of TC to VC mapping.

A Virtual Channel is established when one or multiple TCs are associated with a physical VC resource designated by VC ID. This process is controlled by configuration software as described in Sections 6.3, 7.11, and 7.18.

Support for TCs and VCs beyond default TC0/VC0 pair is optional. The association of TC0 with VC0 is fixed, i.e., "hardwired," and must be supported by all components. Therefore the baseline TC/VC setup does not require any VC-specific hardware or software configuration. In order to ensure interoperability, components that do not implement the optional Virtual Channel Capability structure or Multi-Function Virtual Channel Capability structure must obey the following rules:

- A Requester must only generate requests with TC0 label. (Note that if the Requester initiates requests with a TC label other than TC0, the requests may be treated as malformed by the component on the other side of the Link that implements the extended VC capability and applies TC filtering.)
- A Completer must accept requests with TC label other than TC0, and must preserve the TC label, i.e., any completion that it generates must have the same TC label as the label of the request.
- A Switch must map all TCs to VC0 and must forward all transactions regardless of the TC label.

A Device containing Functions capable of generating Requests with TC labels other than TC0 must implement suitable VC or MFVC Capability structures (as applicable), even if it only supports the default VC. Example Function types are Endpoints and Root Ports. This is required in order to enable mapping of TCs beyond the default configuration. It must follow the TC/VC mapping rules according to the software programming of the VC and MFVC Capability structures.

Figure 2-40 illustrates the concept of Virtual Channel. Conceptually, traffic that flows through VCs is multiplexed onto a common physical Link resource on the Transmit side and de-multiplexed into separate VC paths on the Receive side.

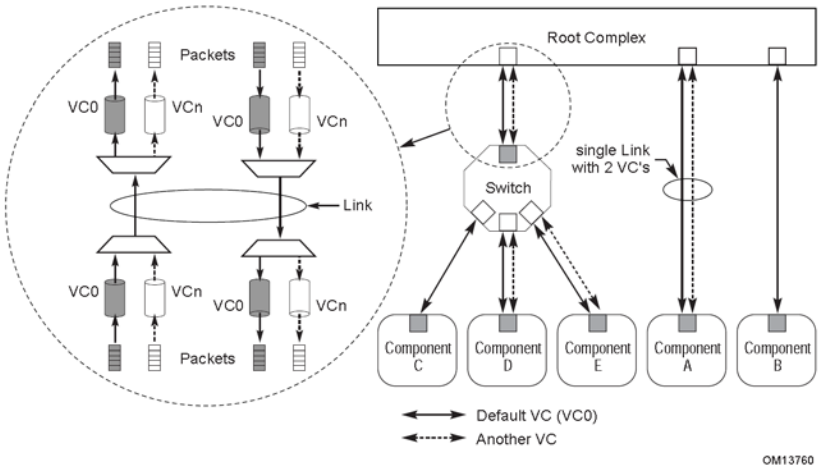


Figure 2-40: Virtual Channel Concept – An Illustration

Internal to the Switch, every Virtual Channel requires dedicated physical resources (queues/buffers and control logic) that support independent traffic flows inside the Switch. Figure 2-41 shows conceptually the VC resources within the Switch (shown in Figure 2-40) that are required to support traffic flow in the Upstream direction.

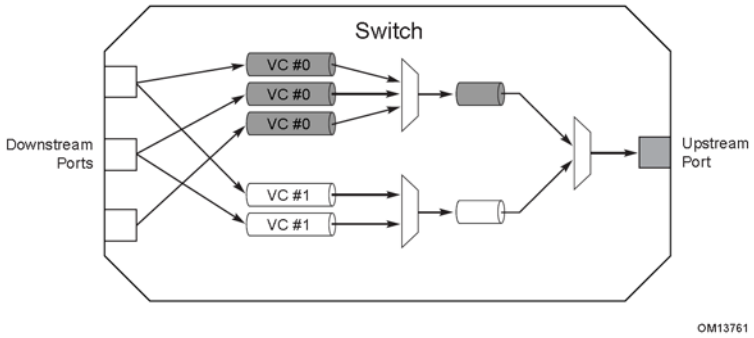


Figure 2-41: Virtual Channel Concept – Switch Internals (Upstream Flow)

A multi-Function device may implement Virtual Channel resources similar to a subset of those in a Switch, for the purpose of managing the QoS for Upstream requests from the different Functions to the device's Upstream Egress Port.



IMPLEMENTATION NOTE

VC and VC Buffering Considerations

The amount of buffering beyond the architectural minimums per supported VC is implementation-specific.

Buffering beyond the architectural minimums is not required to be identical across all VCs on a given Link, i.e., an implementation may provide greater buffer depth for selected VCs as a function of implementation usage models and other Link attributes, e.g., Link width and signaling.

Implementations may adjust their buffering per VC based on implementation-specific policies derived from configuration and VC enablement, e.g., if a four VC implementation has only two VCs enabled, the implementation may assign the non-enabled VC buffering to the enabled VCs to improve fabric efficiency/performance by reducing the probability of fabric backpressure due to Link-level flow control.

The number of VCs supported, and the associated buffering per VC per Port, are not required to be the same for all Ports of a multi-Port component (a Switch or Root Complex).

2.5.1. Virtual Channel Identification (VC ID)

PCI Express Ports can support 1-8 Virtual Channels – each Port is independently configured/managed therefore allowing implementations to vary the number of VCs supported per Port based on usage model-specific requirements. These VCs are uniquely identified using the Virtual Channel Identification (VC ID) mechanism.

Note that while DLLPs contain VC ID information for Flow Control accounting, TLPs do not. The association of TLPs with VC ID for the purpose of Flow Control accounting is done at each Port of the Link using TC to VC mapping as discussed in Section 2.5.2.

All Ports that support more than VC0 must provide at least one VC Capability structure according to the definition in Section 7.11. A multi-Function device is permitted to implement the MFVC Capability structure, as defined in Section 7.18. Providing these extended structures are optional for Ports that support only the default TC0/VC0 configuration. Configuration software is responsible for configuring Ports on both sides of the Link for a matching number of VCs. This is accomplished by scanning the hierarchy and using VC or MFVC Capability registers associated with Ports (that support more than default VC0) to establish the number of VCs for the Link. Rules for assigning VC ID to VC hardware resources within a Port are as follows:

- VC ID assignment must be unique per Port – The same VC ID cannot be assigned to different VC hardware resources within the same Port.
- VC ID assignment must be the same (matching in the terms of numbers of VCs and their IDs) for the two Ports on both sides of a Link.
- If a multi-Function device implements an MFVC Capability structure, its VC hardware resources are distinct from the VC hardware resources associated with any VC Capability structures of its Functions. The VC ID uniqueness requirement (first bullet above) still applies individually for the MVFC and any VC Capability structures. In addition, the VC ID cross-Link matching

requirement (second bullet above) applies for the MFVC Capability structure, but not the VC Capability structures of the Functions.

- VC ID 0 is assigned and fixed to the default VC.

2.5.2. TC to VC Mapping

Every Traffic Class that is supported must be mapped to one of the Virtual Channels. The mapping of TC0 to VC0 is fixed.

The mapping of TCs other than TC0 is system software specific. However, the mapping algorithm must obey the following rules:

- One or multiple TCs can be mapped to a VC.
- One TC must not be mapped to multiple VCs in any Port or Endpoint Function.
- TC/VC mapping must be identical for Ports on both sides of a Link.

Table 2-40 provides an example of TC to VC mapping.

Table 2-40: TC to VC Mapping Example

Supported VC Configurations	TC/VC Mapping Options
VC0	TC(0-7)/VC0
VC0, VC1	TC(0-6)/VC0, TC7/VC1
VC0-VC3	TC(0-1)/VC0, TC(2-4)/VC1, TC(5-6)/VC2, TC7/VC3
VC0-VC7	TC[0:7]/VC[0:7]

Notes on conventions:

- TCn/VCk = TCn mapped to VCk
- TC(n-m)/VCk = all TCs in the range n-m mapped to VCk (i.e., to the same VC)
- TC[n:m]/VC[n:m] = TCn/VCn, TCn_{+1}/ $VCn_{+1,}$..., TCm/VCm

Figure 2-42 provides a graphical illustration of TC to VC mapping in several different Link configurations. For additional considerations on TC/VC, refer to Section 6.3.

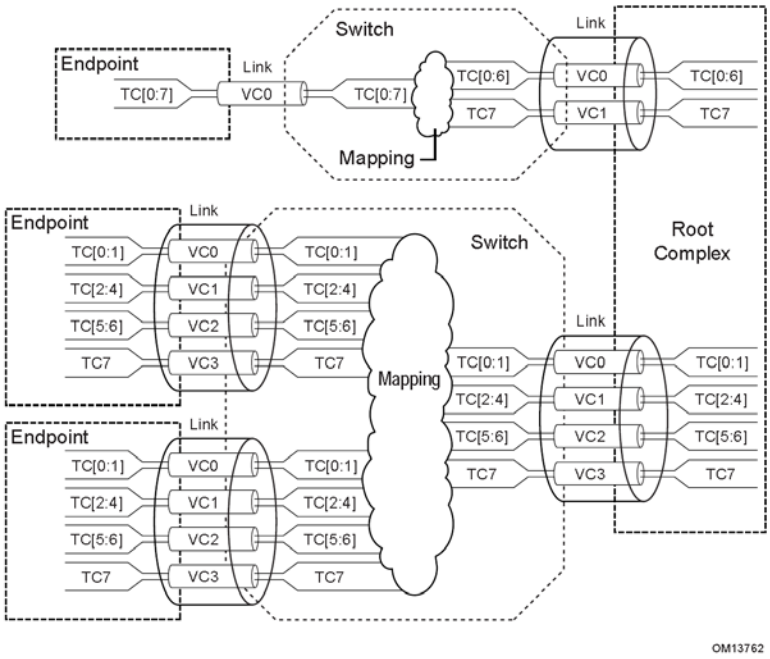


Figure 2-42: An Example of TC/VC Configurations

2.5.3. VC and TC Rules

Here is a summary of key rules associated with the TC/VC mechanism:

- All devices must support the general purpose I/O Traffic Class, i.e., TC0 and must implement the default VC0.
- Each Virtual Channel (VC) has independent Flow Control.
- There are no ordering relationships required between different TCs
- There are no ordering relationships required between different VCs
- A Switch's peer-to-peer capability applies to all Virtual Channels supported by the Switch.
- A multi-Function device's peer-to-peer capability between different Functions applies to all Virtual Channels supported by the multi-Function device.
- Transactions with a TC that is not mapped to any enabled VC in an Ingress Port are treated as Malformed TLPs by the receiving device.
- For Switches, transactions with a TC that is not mapped to any of the enabled VCs in the target Egress Port are treated as Malformed TLPs.

- For a Root Port, transactions with a TC that is not mapped to any of the enabled VCs in the target RCRB are treated as Malformed TLPs.
- For multi-Function devices with an MFVC Capability structure, any transaction with a TC that is not mapped to an enabled VC in the MFVC Capability structure is treated as a Malformed TLP.
- Switches must support independent TC/VC mapping configuration for each Port.
- A Root Complex must support independent TC/VC mapping configuration for each RCRB, the associated Root Ports, and any Root Complex Integrated Endpoints.

For more details on the VC and TC mechanisms, including configuration, mapping, and arbitration, refer to Section 6.3.

2.6. Ordering and Receive Buffer Flow Control

Flow Control (FC) is used to prevent overflow of Receiver buffers and to enable compliance with the ordering rules defined in Section 2.4. Note that the Flow Control mechanism is used by the Requester to track the queue/buffer space available in the agent across the Link as shown in Figure 2-43. That is, Flow Control is point-to-point (across a Link) and not end-to-end. Flow Control does not imply that a Request has reached its ultimate Completer.

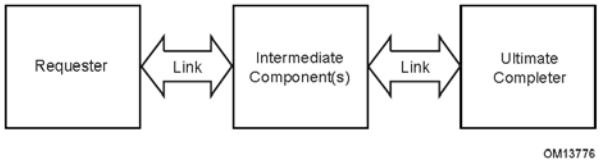


Figure 2-43: Relationship Between Requester and Ultimate Completer

Flow Control is orthogonal to the data integrity mechanisms used to implement reliable information exchange between Transmitter and Receiver. Flow Control can treat the flow of TLP information from Transmitter to Receiver as perfect, since the data integrity mechanisms ensure that corrupted and lost TLPs are corrected through retransmission (see Section 3.5).

Each Virtual Channel maintains an independent Flow Control credit pool. The FC information is conveyed between two sides of the Link using DLLPs. The VC ID field of the DLLP is used to carry the Virtual Channel Identification that is required for proper flow-control credit accounting.

Flow Control mechanisms used internally within a multi-Function device are outside the scope of this specification.

Flow Control is handled by the Transaction Layer in cooperation with the Data Link Layer. The Transaction Layer performs Flow Control accounting functions for Received TLPs and "gates" TLP Transmissions based on available credits for transmission.

Note: Flow Control is a function of the Transaction Layer and, therefore, the following types of information transmitted on the interface are not associated with Flow Control Credits: LCRC, Packet Framing Symbols, other Special Symbols, and Data Link Layer to Data Link Layer inter-communication packets. An implication of this fact is that these types of information must be processed by the Receiver at the rate they arrive (except as explicitly noted in this specification).

Also, any TLPs transferred from the Transaction Layer to the Data Link and Physical Layers must have first passed the Flow Control "gate." Thus, both Transmit and Receive Flow Control mechanisms are unaware if the Data Link Layer transmits a TLP repeatedly due to errors on the Link.

2.6.1. Flow Control Rules

In this and other sections of this specification, rules are described using conceptual "registers" that a device could use in order to implement a compliant implementation. This description does not imply or require a particular implementation and is used only to clarify the requirements.

- Flow Control information is transferred using Flow Control Packets (FCPs), which are a type of DLLP (see Section 3.4)
- The unit of Flow Control credit is 4 DW for data
- For headers:
 - The unit of Flow Control credit for Receivers that do not support TLP Prefixes is the sum of one maximum-size Header and TLP Digest.
 - The unit of Flow Control credits for Receivers that support End-End TLP Prefixes is the sum of one maximum-size Header, TLP Digest, and the maximum number of End-End TLP Prefixes permitted in a TLP.
 - The management of Flow Control for Receivers that support Local TLP Prefixes is dependent on the Local TLP Prefix type.
- Each Virtual Channel has independent Flow Control
- Flow Control distinguishes three types of TLPs (note relationship to ordering rules – see Section 2.4):
 - Posted Requests (P) – Messages and Memory Writes
 - Non-Posted Requests (NP) – All Reads, I/O Writes, Configuration Writes, and AtomicOps
 - Completions (CPL) – Associated with corresponding NP Requests
- In addition, Flow Control distinguishes the following types of TLP information within each of the three types:
 - Headers (H)
 - Data (D)

- Thus, there are six types of information tracked by Flow Control for each Virtual Channel, as shown in Table 2-41.

Table 2-41: Flow Control Credit Types

Credit Type	Applies to This Type of TLP Information
PH	Posted Request headers
PD	Posted Request Data payload
NPH	Non-Posted Request headers
NPD	Non-Posted Request Data payload
CPLH	Completion headers
CPLD	Completion Data payload

- TLPs consume Flow Control credits as shown in Table 2-42.

Table 2-42: TLP Flow Control Credit Consumption

TLP	Credit Consumed[30]
Memory, I/O, Configuration Read Request	1 NPH unit
Memory Write Request	1 PH + n PD units[31]
I/O, Configuration Write Request	1 NPH + 1 NPD Note: size of data written is never more than 1 (aligned) DW
AtomicOp Request	1 NPH + n NPD units
Message Requests without data	1 PH unit
Message Requests with data	1 PH + n PD units
Memory Read Completion	1 CPLH + n CPLD units
I/O, Configuration Read Completions	1 CPLH unit + 1 CPLD unit
I/O, Configuration Write Completions	1 CPLH unit
AtomicOp Completion	1 CPLH unit + 1 CPLD unit Note: size of data returned is never more than 4 (aligned) DWs.

- Components must implement independent Flow Control for all Virtual Channels that are supported by that component.
- Flow Control is initialized autonomously by hardware only for the default Virtual Channel (VC0)
 - VC0 is initialized when the Data Link Layer is in the DL_Init state following reset (see Sections 3.2 and 3.3)

[30] Each header credit implies the ability to accept a TLP Digest along with the corresponding TLP.

[31] For all cases where "n" appears, n = Roundup(Length/FC unit size).

- When other Virtual Channels are enabled by software, each newly enabled VC will follow the Flow Control initialization protocol (see Section 3.3)
 - Software enables a Virtual Channel by setting the VC Enable bits for that Virtual Channel in both components on a Link (see Sections 7.11 and 7.18)

Note: It is possible for multiple VCs to be following the Flow Control initialization protocol simultaneously – each follows the initialization protocol as an independent process

- Software disables a Virtual Channel by clearing the VC Enable bits for that Virtual Channel in both components on a Link
 - Disabling a Virtual Channel for a component resets the Flow Control tracking mechanisms for that Virtual Channel in that component
- InitFC1 and InitFC2 FCPs are used only for Flow Control initialization (see Section 3.3)
- An InitFC1, InitFC2, or UpdateFC FCP that specifies a Virtual Channel that is disabled is discarded without effect
- During FC initialization for any Virtual Channel, including the default VC initialized as a part of Link initialization, Receivers must initially advertise VC credit values equal to or greater than those shown in Table 2-43.

Table 2-43: Minimum Initial Flow Control Advertisements[32]

Credit Type	Minimum Advertisement
PH	1 unit – credit value of 01h
PD	Largest possible setting of the Max_Payload_Size for the component divided by FC Unit Size. For a multi-Function device, this includes all Functions in the device. Example: If the largest Max_Payload_Size value supported is 1024 bytes, the smallest permitted initial credit value would be 040h.
NPH	1 unit – credit value of 01h
NPD	Receiver that supports AtomicOp routing capability or any AtomicOp Completer capability: 2 units – credit value of 002h All other Receivers: 1 unit – credit value of 001h
CPLH	Root Complex (supporting peer-to-peer traffic between all Root Ports) and Switch: 1 FC unit - credit value of 01h Root Complex (not supporting peer-to-peer traffic between all Root Ports) and Endpoint: infinite FC units - initial credit value of all 0s[33]

[32] PCI Express to PCI/PCI-X Bridge requirements are addressed in the *PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0*.

[33] This value is interpreted as infinite by the Transmitter, which will, therefore, never throttle.

Credit Type	Minimum Advertisement
CPLD	Root Complex (supporting peer-to-peer traffic between all Root Ports) and Switch: Largest possible setting of the Max_Payload_Size for the component divided by FC Unit Size., Root Complex (not supporting peer-to-peer traffic between all Root Ports) and Endpoint: infinite FC units - initial credit value of all 0s

- A Root Complex that ~~does not~~ supports no peer-to-peer traffic between ~~all~~ Root Ports must advertise infinite Completion credits on every Root Port.
- A Root Complex that supports peer-to-peer traffic between some or all of its Root Ports may optionally advertise non-infinite Completion credits on those Root Ports. In this case, the Root Complex must ensure that deadlocks are avoided and forward progress is maintained for completions directed towards the Root Complex. Note that temporary stalls of completion traffic (due to a temporary lack of credit) are possible since Non-Posted requests forwarded by the RC may not have explicitly allocated completion buffer space.
- A Receiver must never cumulatively issue more than 2047 outstanding unused credits to the Transmitter for data payload or 127 for header.
 - Components may optionally check for violations of this rule. If a component implementing this check determines a violation of this rule, the violation is a Flow Control Protocol Error (FCPE).
 - If checked, this is a reported error associated with the Receiving Port (see Section 6.2)
- If an Infinite Credit advertisement (value of 00h or 000h) has been made during initialization, no Flow Control updates are required following initialization.
 - If UpdateFC DLLPs are sent, the credit value fields must be set to zero and must be ignored by the Receiver. The Receiver may optionally check for non-zero update values (in violation of this rule). If a component implementing this check determines a violation of this rule, the violation is a Flow Control Protocol Error (FCPE)
 - If checked, this is a reported error associated with the Receiving Port (see Section 6.2)
- If only the Data or header advertisement (but not both) for a given type (P, NP, or CPL) has been made with infinite credits during initialization, the transmission of UpdateFC DLLPs is still required, but the credit field corresponding to the Data/header (advertised as infinite) must be set to zero and must be ignored by the Receiver.
 - The Receiver may optionally check for non-zero update values (in violation of this rule). If a Receiver implementing this check determines a violation of this rule, the violation is a Flow Control Protocol Error (FCPE)
 - If checked, this is a reported error associated with the Receiving Port (see Section 6.2)

Comment [VS82]: Errata B22

Comment [VS83]: Errata B22

- A received TLP using a VC that is not enabled is a Malformed TLP
 - VC0 is always enabled
 - For VCs 1-7, a VC is considered enabled when the corresponding VC Enable bit in the VC Resource Control register has been set to 1b, and once FC negotiation for that VC has exited the FC_INIT1 state and progressed to the FC_INIT2 state (see Section 3.3)
 - This is a reported error associated with the Receiving Port (see Section 6.2)
- TLP transmission using any VC 0-7 is not permitted until initialization for that VC has completed by exiting FC_INIT2 state

For VCs 1-7, software must use the VC Negotiation Pending bit in the VC Resource Status register to ensure that a VC is not used until negotiation has completed by exiting the FC_INIT2 state in both components on a Link.

2.6.1.1. FC Information Tracked by Transmitter

- For each type of information tracked, there are two quantities tracked for Flow Control TLP Transmission gating:
 - CREDITS_CONSUMED
 - Count of the total number of FC units consumed by TLP Transmissions made since Flow Control initialization, modulo $2^{[\text{Field Size}]}$ (where [Field Size] is 8 for PH, NPH, and CPLH and 12 for PD, NPD and CPLD).
 - Set to all 0's at interface initialization
 - Updated for each TLP the Transaction Layer allows to pass the Flow Control gate for Transmission
 - Updated as shown:

 CREDITS_CONSUMED :=

 (CREDITS_CONSUMED + Increment) mod $2^{[\text{Field Size}]}$

 Where Increment is the size in FC credits of the corresponding part of the TLP passed through the gate, and [Field Size] is 8 for PH, NPH, and CPLH and 12 for PD, NPD, and CPLD
 - CREDIT_LIMIT
 - The most recent number of FC units legally advertised by the Receiver. This quantity represents the total number of FC credits made available by the Receiver since Flow Control initialization, , modulo $2^{[\text{Field Size}]}$ (where [Field Size] is 8 for PH, NPH, and CPLH and 12 for PD, NPD, and CPLD).
 - Undefined at interface initialization
 - Set to the value indicated during Flow Control initialization
 - For each FC update received,

- if CREDIT_LIMIT is not equal to the update value, set CREDIT_LIMIT to the update value

- If a Transmitter detects that a TLP it is preparing to transmit is malformed, it is strongly recommended that the Transmitter discard the TLP and handle the condition as an Uncorrectable Internal Error.
- If a Transmitter detects that a TLP it is preparing to transmit appears to be properly formed but with bad ECRC, it is strongly recommended that the Transmitter transmit the TLP and update its internal Flow Control credits accordingly.

Comment [VS84]: Errata A24 (Part C list resolved 10/13)

Comment [VS85]: Errata A24 (Part C list resolved 10/13)

- The Transmitter gating function must determine if sufficient credits have been advertised to permit the transmission of a given TLP. If the Transmitter does not have enough credits to transmit the TLP, it must block the transmission of the TLP, possibly stalling other TLPs that are using the same Virtual Channel. The Transmitter must follow the ordering and deadlock avoidance rules specified in Section 2.4, which require that certain types of TLPs must bypass other specific types of TLPs when the latter are blocked. Note that TLPs using different Virtual Channels have no ordering relationship, and must not block each other.
- The Transmitter gating function test is performed as follows:
 - For each required type of credit, the number of credits required is calculated as:

 CUMULATIVE_CREDITS_REQUIRED =

 (CREDITS_CONSUMED + <credit units required for pending TLP>) mod $2^{[\text{Field Size}]}$
 - Unless CREDIT_LIMIT was specified as "infinite" during Flow Control initialization, the Transmitter is permitted to Transmit a TLP if, for each type of information in the TLP, the following equation is satisfied (using unsigned arithmetic):

 (CREDIT_LIMIT – CUMULATIVE_CREDITS_REQUIRED) mod $2^{[\text{Field Size}]}$

 <= $2^{[\text{Field Size}]}$ / 2

 If CREDIT_LIMIT was specified as "infinite" during Flow Control initialization, then the gating function is unconditionally satisfied for that type of credit.

 Note that some types of Transactions require more than one type of credit. (For example, Memory Write requests require PH and PD credits.)
- When accounting for credit use and return, information from different TLPs is never mixed within one credit.
- When some TLP is blocked from Transmission by a lack of FC Credit, Transmitters must follow the ordering rules specified in Section 2.4 when determining what types of TLPs must be permitted to bypass the stalled TLP.

- The return of FC credits for a Transaction must not be interpreted to mean that the Transaction has completed or achieved system visibility.
 - Flow Control credit return is used for receive buffer management only, and agents must not make any judgment about the Completion status or system visibility of a Transaction based on the return or lack of return of Flow Control information.
- When a Transmitter sends a nullified TLP, the Transmitter does not modify CREDITS_CONSUMED for that TLP (see Section 3.5.2.1).

2.6.1.2. FC Information Tracked by Receiver

- For each type of information tracked, the following quantities are tracked for Flow Control TLP Receiver accounting:
 - CREDITS_ALLOCATED
 - Count of the total number of credits granted to the Transmitter since initialization, modulo $2^{[\text{Field Size}]}$ (where [Field Size] is 8 for PH, NPH, and CPLH and 12 for PD, NPD and CPLD)
 - Initially set according to the buffer size and allocation policies of the Receiver
 - This value is included in the InitFC and UpdateFC DLLPs (see Section 3.4)
 - Incremented as the Receiver Transaction Layer makes additional receive buffer space available by processing Received TLPs
 - Updated as shown:

 CREDITS_ALLOCATED :=

 CREDITS_ALLOCATED + Increment) mod $2^{[\text{Field Size}]}$

 Where Increment corresponds to the credits made available, and [Field Size] is 8 for PH, NPH, and CPLH and 12 for PD, NPD, and CPLD
 - CREDITS_RECEIVED (Optional – for optional error check described below)
 - Count of the total number of FC units consumed by valid TLPs Received since Flow Control initialization, modulo $2^{[\text{Field Size}]}$ (where [Field Size] is 8 for PH, NPH, and CPLH and 12 for PD, NPD and CPLD)
 - Set to all 0's at interface initialization
 - Updated as shown:

 CREDITS_RECEIVED :=

 (CREDITS_ RECEIVED + Increment) mod $2^{[\text{Field Size}]}$

 (Where Increment ~~corresponds to the credits made available~~ is the size in FC units of the corresponding part of the received TLP, and [Field Size] is 8 for PH, NPH, and CPLH and 12 for PD, NPD, and CPLD)

 for each Received TLP, provided that TLP:
 - passes the Data Link Layer integrity checks

Comment [VS86]: Errata B10

- is not malformed or (optionally) is malformed and is not ambiguous with respect to which buffer to release and is mapped to an initialized Virtual Channel
- does not consume more credits than have been allocated (see following rule)

For a TLP with an ECRC Check Failed error, but which otherwise is unambiguous with respect to which buffer to release, it is strongly recommended that CREDITS_RECEIVED be updated.

Comment [VS87]: Errata A24 (Part C list resolved 10/13)

- ❑ If a Receiver implements the CREDITS_RECEIVED counter, then when a nullified TLP is received, the Receiver does not modify CREDITS_RECEIVED for that TLP (see Section 3.5.2.1)
- ❑ A Receiver may optionally check for Receiver Overflow errors (TLPs exceeding CREDITS_ALLOCATED), by checking the following equation, using unsigned arithmetic:

 $$(\text{CREDITS_ALLOCATED} - \text{CREDITS_RECEIVED}) \bmod 2^{[\text{Field Size}]} \geq 2^{[\text{Field Size}]} / 2$$

 If the check is implemented and this equation evaluates as true, the Receiver must:
 - o discard the TLP(s) without modifying the CREDITS_RECEIVED
 - o de-allocate any resources which it had allocated for the TLP(s)

 If checked, this is a reported error associated with the Receiving Port (see Section 6.2)

 Note: Following a Receiver Overflow error, Receiver behavior is undefined, but it is encouraged that the Receiver continues to operate, processing Flow Control updates and accepting any TLPs which do not exceed allocated credits.
- ❑ For non-infinite NPH, NPD, PH, and CPLH types, an UpdateFC FCP must be scheduled for Transmission each time the following sequence of events occurs:
 - o (a) when the number of available FC credits of a particular type is zero or (b) the NPD credit drops below 2 and the Receiver supports AtomicOp routing capability or 128-bit CAS Completer capability
 - o one or more units of that type are made available by TLPs processed
- ❑ For non-infinite PD and CPLD types, when the number of available credits is less than Max_Payload_Size, an UpdateFC FCP must be scheduled for Transmission each time one or more units of that type are made available by TLPs processed
 - o For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. The Max_Payload_Size settings in other Functions are ignored.
 - o For a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the common Max_Payload_Size setting or larger must be used.
 - o For a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the selected Max_Payload_Size setting is implementation specific, but it is recommended to use the largest Max_Payload_Size setting across all Functions.
- ❑ UpdateFC FCPs may be scheduled for Transmission more frequently than is required

- When the Link is in the L0 or L0s Link state, Update FCPs for each enabled type of non-infinite FC credit must be scheduled for transmission at least once every 30 µs (-0%/+50%), except when the Extended Sync bit of the Control Link register is set, in which case the limit is 120 µs (-0%/+50%).
 - A timeout mechanism may optionally be implemented. If implemented, such a mechanism must:
 - be active only when the Link is in the L0 or L0s Link state
 - use a timer with a limit of 200 µs (-0%/+50%), where the timer is reset by the receipt of any Init or Update FCP. Alternately, the timer may be reset by the receipt of any DLLP (see Section 3.4)
 - upon timer expiration, instruct the Physical Layer to retrain the Link (via the LTSSM Recovery state, Section 4.2.6.4)
 - if an Infinite Credit advertisement has been made during initialization for all three Flow Control classes, this timeout mechanism must be disabled

Note: The implementation of this optional mechanism is strongly encouraged. Future revisions of this specification may change this mechanism from optional to required.



IMPLEMENTATION NOTE

Use of "Infinite" FC Advertisement

For a given implementation it is possible that not all of the queue types need to be physically implemented in hardware for all Virtual Channels. For example, ~~since Non-Posted Writes are only allowed on Virtual Channel 0~~in a Device whose Functions have no AtomicOp Completer or AtomicOp Routing capability, there is no need to implement a Non-Posted Data queue for Virtual Channels other than VC0, since Non-Posted Requests with data are only allowed on Virtual Channel 0 for such Devices. For unimplemented queues, the Receiver can eliminate the need to present the appearance of tracking Flow Control credits by advertising infinite Flow Control credits during initialization.

Comment [VS88]: Errata B28

IMPLEMENTATION NOTE

Flow Control Update Frequency

For components subject to receiving streams of TLPs, it is desirable to implement receive buffers larger than the minimum size required to prevent Transmitter throttling due to lack of available credits. Likewise, UpdateFC FCPs must be returned such that the time required to send, receive and process the UpdateFC is sufficient. Recommended values for the frequency of transmission during normal operation based on Link Width and Max_Payload_Size values for operation in 2.5 GT/s mode are shown in Table 2-44. for operation in 5.0 GT/s mode are shown in Table 2-45, and for operation in 8.0 GT/s mode are shown in Table 2-46. Note that the values given in Table 2-44, Table 2-45, and Table 2-46 do not account for any delays caused by the Receiver or Transmitter being in L0s. For improved performance and/or power-saving, it may be desirable to use a Flow Control update policy that is more sophisticated than a simple timer. Any such policy is implementation specific, and beyond the scope of this document.

The values are calculated as a function of the largest TLP payload size and Link width. The values are measured at the Port of the TLP Receiver, starting with the time the last Symbol of a TLP is received to the first Symbol of the UpdateFC DLLP being transmitted. The values are calculated using the formula:

$$\frac{(Max_Payload_Size + TLPOverhead) * UpdateFactor}{LinkWidth} + InternalDelay$$

where:

Max_Payload_Size	The value in the Max_Payload_Size field of the Device Control register. For a multi-Function device, it is recommended that the smallest Max_Payload_Size setting across all Functions[34] is used.
TLPOverhead	Represents the additional TLP components which consume Link bandwidth (TLP Prefix, header, LCRC, framing Symbols) and is treated here as a constant value of 28 Symbols.
UpdateFactor	Represents the number of maximum size TLPs sent during the time between UpdateFC receptions, and is used to balance Link bandwidth efficiency and receive buffer sizes – the value varies according to Max_Payload_Size and Link width, and is included in Table 2-44, Table 2-45, and Table 2-46.
LinkWidth	The operating width of the Link
InternalDelay	Represents the internal processing delays for received TLPs and transmitted DLLPs, and is treated here as a constant value of 19 Symbol Times for 2.5 GT/s mode operation, 70 Symbol Times for 5.0 GT/s mode operation, and 115 Symbol Times for 8.0 GT/s mode operation.

[34] For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0, and thus the settings in the other Functions should be ignored.

Table 2-44: UpdateFC Transmission Latency Guidelines for 2.5 GT/s Mode Operation by Link Width and Max Payload (Symbol Times)

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	237 UF = 1.4	128 UF = 1.4	73 UF = 1.4	67 UF = 2.5	58 UF = 3.0	48 UF = 3.0	33 UF = 3.0
	256	416 UF = 1.4	217 UF = 1.4	118 UF = 1.4	107 UF = 2.5	90 UF = 3.0	72 UF = 3.0	45 UF = 3.0
	512	559 UF = 1.0	289 UF = 1.0	154 UF = 1.0	86 UF = 1.0	109 UF = 2.0	86 UF = 2.0	52 UF = 2.0
	1024	1071 UF = 1.0	545 UF = 1.0	282 UF = 1.0	150 UF = 1.0	194 UF = 2.0	150 UF = 2.0	84 UF = 2.0
	2048	2095 UF = 1.0	1057 UF = 1.0	538 UF = 1.0	278 UF = 1.0	365 UF = 2.0	278 UF = 2.0	148 UF = 2.0
	4096	4143 UF = 1.0	2081 UF = 1.0	1050 UF = 1.0	534 UF = 1.0	706 UF = 2.0	534 UF = 2.0	276 UF = 2.0

Table 2-45: UpdateFC Transmission Latency Guidelines for 5.0 GT/s Mode Operation by Link Width and Max Payload (Symbol Times)

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	288 UF = 1.4	179 UF = 1.4	124 UF = 1.4	118 UF = 2.5	109 UF = 3.0	99 UF = 3.0	84 UF = 3.0
	256	467 UF = 1.4	268 UF = 1.4	169 UF = 1.4	158 UF = 2.5	141 UF = 3.0	123 UF = 3.0	96 UF = 3.0
	512	610 UF = 1.0	340 UF = 1.0	205 UF = 1.0	137 UF = 1.0	160 UF = 2.0	137 UF = 2.0	103 UF = 2.0
	1024	1122 UF = 1.0	596 UF = 1.0	333 UF = 1.0	201 UF = 1.0	245 UF = 2.0	201 UF = 2.0	135 UF = 2.0
	2048	2146 UF = 1.0	1108 UF = 1.0	589 UF = 1.0	329 UF = 1.0	416 UF = 2.0	329 UF = 2.0	199 UF = 2.0
	4096	4194 UF = 1.0	2132 UF = 1.0	1101 UF = 1.0	585 UF = 1.0	757 UF = 2.0	585 UF = 2.0	327 UF = 2.0

Table 2-46: UpdateFC Transmission Latency Guidelines for 8.0 GT/s Operation by Link Width and Max Payload (Symbol Times)

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	333 UF = 1.4	224 UF = 1.4	169 UF = 1.4	163 UF = 2.5	154 UF = 3.0	144 UF = 3.0	129 UF = 3.0
	256	512 UF = 1.4	313 UF = 1.4	214 UF = 1.4	203 UF = 2.5	186 UF = 3.0	168 UF = 3.0	141 UF = 3.0
	512	655 UF = 1.0	385 UF = 1.0	250 UF = 1.0	182 UF = 1.0	205 UF = 2.0	182 UF = 2.0	148 UF = 2.0
	1024	1167 UF = 1.0	641 UF = 1.0	378 UF = 1.0	246 UF = 1.0	290 UF = 2.0	246 UF = 2.0	180 UF = 2.0
	2048	2191 UF = 1.0	1153 UF = 1.0	634 UF = 1.0	374 UF = 1.0	461 UF = 2.0	374 UF = 2.0	244 UF = 2.0
	4096	4239 UF = 1.0	2177 UF = 1.0	1146 UF = 1.0	630 UF = 1.0	802 UF = 2.0	630 UF = 2.0	372 UF = 2.0

2.7. Data Integrity

The basic data reliability mechanism in PCI Express is contained within the Data Link Layer, which uses a 32-bit CRC (LCRC) code to detect errors in TLPs on a Link-by-Link basis, and applies a Link-by-Link retransmit mechanism for error recovery. A TLP is a unit of data and transaction control that is created by a data-source at the "edge" of the PCI Express domain (such as an Endpoint or Root Complex), potentially routed through intermediate components (i.e., Switches) and consumed by the ultimate PCI Express recipient. As a TLP passes through a Switch, the Switch may need to change some control fields without modifying other fields that should not change as the packet traverses the path. Therefore, the LCRC is regenerated by Switches. Data corruption may occur internally to the Switch, and the regeneration of a good LCRC for corrupted data masks the existence of errors. To ensure end-to-end data integrity detection in systems that require high data reliability, a Transaction Layer end-to-end 32-bit CRC (ECRC) can be placed in the TLP Digest field at the end of a TLP. The ECRC covers all fields that do not change as the TLP traverses the path (invariant fields). The ECRC is generated by the Transaction Layer in the source component, and checked (if supported) by the ultimate PCI Express Receiver and optionally by intermediate Receivers. A Switch that supports ECRC checking must check ECRC on TLPs targeting the Switch itself. Such a Switch can optionally check ECRC on TLPs that it forwards. On TLPs that the Switch forwards, the Switch must preserve the ECRC (forward it untouched) as an integral part of the TLP, regardless of whether the Switch checks the ECRC or if the ECRC check fails.[35]

In some cases, the data in a TLP payload is known to be corrupt at the time the TLP is generated, or may become corrupted while passing through an intermediate component, such as a Switch. In these cases, error forwarding, also known as data poisoning, can be used to indicate the corruption to the device consuming the data.

2.7.1. ECRC Rules

The capability to generate and check ECRC is reported to software, and the ability to do so is enabled by software (see Section 7.10.7).

- If a device Function is enabled to generate ECRC, it must calculate and apply ECRC for all TLPs originated by the Function
- Switches must pass TLPs with ECRC unchanged from the Ingress Port to the Egress Port[35]
- If a device supports ECRC generation/checking, at least one of its Functions must support Advanced Error Reporting (see Section 6.2)
- If a device Function is enabled to check ECRC, it must do so for all TLPs with ECRC where the device is the ultimate PCI Express Receiver
 - Note that it is still possible for the Function to receive TLPs without ECRC, and these are processed normally – this is not an error

[35] An exception is a Multicast TLP that an Egress Port is modifying due to the MC_Overlay mechanism. See Section 6.14.5.

Note that a Switch may optionally perform ECRC checking on TLPs passing through the Switch. ECRC Errors detected by the Switch are reported as described in Table 6-5, but do not alter the TLPs' passage through the Switch.[35]

A 32-bit ECRC is calculated for the TLP (End-End TLP Prefixes, header, and data payload) using the following algorithm and appended to the end of the TLP (see Figure 2-3):

- The ECRC value is calculated using the following algorithm (see Figure 2-44)
- The polynomial used has coefficients expressed as 04C1 1DB7h
- The seed value (initial value for ECRC storage registers) is FFFF FFFFh
- All header ~~invariant~~ fields, ~~of the TLP's~~ all End-End TLP Prefixes (if present), ~~header,~~ and the entire data payload (if present) are included in the ECRC calculation.~~,~~ A~~a~~ll bits in variant fields must be set to 1b for ECRC calculations.
 - Bit 0 of the Type field in a TLP header is variant[36]. This bit in an End-End TLP Prefix is invariant.
 - The EP bit is variant
 - all other fields are invariant
- ECRC calculation starts with bit 0 of byte 0 and proceeds from bit 0 to bit 7 of each byte of the TLP
- The result of the ECRC calculation is complemented, and the complemented result bits are mapped into the 32-bit TLP Digest field as shown in Table 2-47.

Comment [VS89]: Errata B30

Comment [VS90]: Errata B8

Table 2-47: Mapping of Bits into ECRC Field

ECRC Result Bit	Corresponding Bit Position in the 32-bit TLP Digest Field
0	7
1	6
2	5
3	4
4	3
5	2
6	1
7	0
8	15
9	14
10	13
11	12
12	11
13	10
14	9

[36] Bit 0 of the Type field changes when a Configuration Request is changed from Type 1 to Type 0.

ECRC Result Bit	Corresponding Bit Position in the 32-bit TLP Digest Field
15	8
16	23
17	22
18	21
19	20
20	19
21	18
22	17
23	16
24	31
25	30
26	29
27	28
28	27
29	26
30	25
31	24

- The 32-bit ECRC value is placed in the TLP Digest field at the end of the TLP (see Figure 2-3)
- For TLPs including a TLP Digest field used for an ECRC value, Receivers which support end-to-end data integrity checking, check the ECRC value in the TLP Digest field by:
 - applying the same algorithm used for ECRC calculation (above) to the received TLP, not including the 32-bit TLP Digest field of the received TLP
 - comparing the calculated result with the value in the TLP Digest field of the received TLP
- Receivers which support end-to-end data integrity checks report violations as an ECRC Error. This reported error is associated with the receiving Port (see Section 6.2).

Beyond the stated error reporting semantics contained elsewhere in this specification, how ~~How~~ ultimate PCI Express Receivers make use of the end-to-end data integrity check provided through the ECRC is beyond the scope of this document. Intermediate Receivers are still required to forward TLPs whose ECRC checks fail. A PCI Express-to-PCI/PCI-X Bridge is classified as an ultimate PCI Express Receiver with regard to ECRC checking.

Comment [VS91]: Errata A24 (Part C list resolved 10/13)

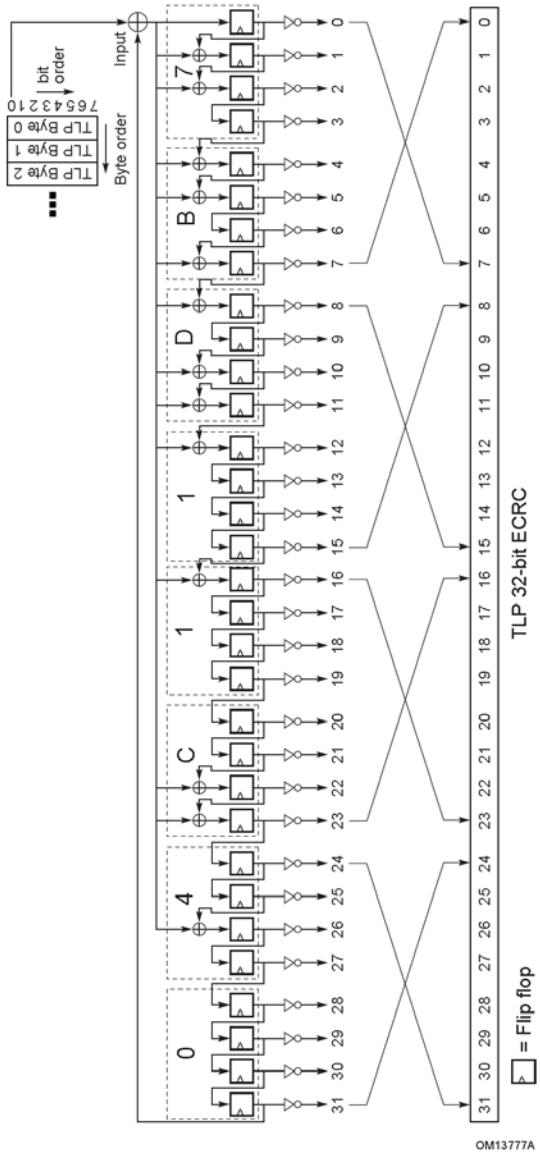


Figure 2-44: Calculation of 32-bit ECRC for TLP End to End Data Integrity Protection

IMPLEMENTATION NOTE

Protection of TD Bit Inside Switches

It is of utmost importance that Switches insure and maintain the integrity of the TD bit in TLPs that they receive and forward (i.e., by applying a special internal protection mechanism), since corruption of the TD bit will cause the ultimate target device to misinterpret the presence or absence of the TLP digest field.

Similarly, it is highly recommended that Switches provide internal protection to other variant fields in TLPs that they receive and forward, as the end-to-end integrity of variant fields is not sustained by the ECRC.

IMPLEMENTATION NOTE

Data Link Layer Does Not Have Internal TLP Visibility

Since the Data Link Layer does not process the TLP header (it determines the start and end of the TLP based on indications from the Physical Layer), it is not aware of the existence of the TLP Digest field, and simply passes it to the Transaction Layer as a part of the TLP.

2.7.2. Error Forwarding

Error Forwarding (also known as data poisoning), is indicated by setting the EP bit to 1b. The rules for doing this are specified in Section 2.7.2.2. Here are some examples of cases where Error Forwarding might be used:

- Example #1: A read from main memory encounters uncorrectable error
- Example #2: Parity error on a PCI write to main memory
- Example #3: Data integrity error on an internal data buffer or cache.

2.7.2.1. Error Forwarding Usage Model

- Error Forwarding is only used for Read Completion Data, AtomicOp Completion Data, AtomicOp Request Data, or Write Data, never for the cases when the error is in the "header" (request phase, address/command, etc.). Requests/Completions with header errors cannot be forwarded in general since true destination cannot be positively known and, therefore, forwarding may cause direct or side effects such as data corruption, system failures, etc.
- Error Forwarding is used for controlled propagation of errors through the system, system diagnostics, etc.

- Note that Error forwarding does not cause Link Layer Retry – Poisoned TLPs will be retried only if there are transmission errors on the Link as determined by the TLP error detection mechanisms in the Data Link Layer.
 - The Poisoned TLP may ultimately cause the originator of the request to re-issue it (at the Transaction Layer or above) in the case of read operation or to take some other action. Such use of Error Forwarding information is beyond the scope of this specification.

2.7.2.2. Rules For Use of Data Poisoning

- Support for TLP poisoning in a Transmitter is optional.
- Data poisoning applies only to the data within a Write Request (Posted or Non-Posted), a Message with Data, an AtomicOp Request, a Read Completion, or an AtomicOp Completion.
 - Poisoning of a TLP is indicated by a 1b value in the EP bit
 - Transmitters are permitted to set the EP bit to 1b only for TLPs that include a data payload. The behavior of the receiver is not specified if the EP bit is set for any TLP that does not include a data payload.
- If a Transmitter supports data poisoning, TLPs that are known to the Transmitter to include bad data must use the poisoning mechanism defined above.
- If a Downstream Port supports Poisoned TLP Egress Blocking, the Poisoned TLP Egress Blocking Enable bit is Set, and a poisoned TLP targets going out the Egress Port, the Port must handle the TLP as a Poisoned TLP Egress Blocked error unless there is a higher precedence error. See Sections 6.2.3.2.3, 6.2.5, and 7.31.2.
 Further:
 - The Port must not transmit the TLP.
 - If the Poisoned TLP Egress Blocked error is unmasked and DPC is enabled, DPC must be triggered and the Port must behave as described in Section 2.9.3.
 - If DPC is not triggered and the TLP is a Non-Posted Request, the Port must return a Completion the same as if the TLP had triggered DPC. See Section 2.9.3.
- The following Requests with Poisoned data must not modify the value of the target location:
 - Configuration Write Request
 - Any of the following that target a control register or control structure in the Completer: I/O Write Request, Memory Write Request, or non-vendor-defined Message with data
 - AtomicOp Request

 Unless there is a higher precedence error, a Completer must handle these Requests as a Poisoned TLP Received error,[37] and the Completer must also return a Completion with a Completion Status of Unsupported Request (UR) if the Request is Non-Posted (see Section 6.2.3.2.3, Section 6.2.3.2.4, and Section 6.2.5). Regardless of the severity of the reported

Comment [VS92]: ECN: Enhanced DPC

[37] Due to ambiguous language in earlier versions of this specification, a component is permitted to handle this error as an Unsupported Request, but this is strongly discouraged.

error, the reported error must be handled as an uncorrectable error, not an Advisory Non-Fatal Error.

Comment [VS93]: Errata A12

A Switch must route the Request the same way it would route the same Request if it were not Poisoned, unless the Request targets a location in the Switch itself, in which case the Switch is the Completer for the Request and must follow the above rules.

For some applications it may be desirable for the Completer to use poisoned data in Write Requests which do not target control registers or control structures – such use is not forbidden. Similarly, it may be desirable for the Requester to use data marked poisoned in Completions – such use is also not forbidden. The appropriate use of poisoned information is application specific, and is not discussed in this document.

This document does not define any mechanism for determining which part or parts of the data payload of a Poisoned TLP are actually corrupt and which, if any, are not corrupt.

2.8. Completion Timeout Mechanism

In any split transaction protocol, there is a risk associated with the failure of a Requester to receive an expected Completion. To allow Requesters to attempt recovery from this situation in a standard manner, the Completion Timeout mechanism is defined. This mechanism is intended to be activated only when there is no reasonable expectation that the Completion will be returned, and should never occur under normal operating conditions. Note that the values specified here do not reflect expected service latencies, and must not be used to estimate typical response times.

PCI Express device Functions that issue Requests requiring Completions must implement the Completion Timeout mechanism. An exception is made for Configuration Requests (see below). The Completion Timeout mechanism is activated for each Request that requires one or more Completions when the Request is transmitted. Since Switches do not autonomously initiate Requests that need Completions, the requirement for Completion Timeout support is limited only to Root Complexes, PCI Express-PCI Bridges, and Endpoints.

The Completion Timeout mechanism may be disabled by configuration software. The Completion Timeout limit is set in the Completion Timeout Value field of the Device Control 2 register. Refer to Section 2.2.8.10. A Completion Timeout is a reported error associated with the Requester Function (see Section 6.2).

Note: A Memory Read Request for which there are multiple Completions must be considered completed only when all Completions have been received by the Requester. If some, but not all, requested data is returned before the Completion Timeout timer expires, the Requester is permitted to keep or to discard the data that was returned prior to timer expiration.

Completion timeouts for Configuration Requests have special requirements for the support of PCI Express to PCI/PCI Express bridges. PCI Express to PCI/PCI-X Bridges, by default, are not enabled to return Configuration Request Retry Status (CRS) for Configuration Requests to a PCI/PCI-X device behind the Bridge. This may result in lengthy completion delays that must be comprehended by the Completion Timeout value in the Root Complex. System software may enable PCI Express to PCI/PCI-X Bridges to return Configuration Request Retry Status by setting the Bridge Configuration Retry Enable bit in the Device Control register, subject to the restrictions noted in the *PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0*.

Formatted: Font color: Black



IMPLEMENTATION NOTE

Completion Timeout Prefix/Header Log Capable

The prefix/header of the Request TLP associated with a Completion Timeout may optionally be recorded by Requesters that implement the Advanced Error Reporting Capability. Support for recording of the prefix/header is indicated by the value of the Completion Timeout Prefix/Header Log Capable bit in the Advanced Error Capabilities and Control register.

A Completion Timeout may be the result of improper configuration, system failure, or Async Removal (see Section 6.7.5). In order for host software to distinguish a Completion Timeout error after which continued normal operation is not possible (e.g., after one caused by improper configuration or a system failure) from one where continued normal operation is possible (e.g., after an Async Removal), it is strongly encouraged that Requesters log the Request TLP prefix/header associated with the Completion Timeout.

Comment [VS94]: ECN: Downstream Port containment (eDPC)

2.9. Link Status Dependencies

2.9.1. Transaction Layer Behavior in DL_Down Status

DL_Down status indicates that there is no connection with another component on the Link, or that the connection with the other component has been lost and is not recoverable by the Physical or Data Link Layers. This section specifies the Transaction Layer's behavior if Downstream Port Containment has not been triggered and ~~when~~ the Data Link Layer reports DL_Down status to the Transaction Layer, indicating that the Link is non-operational. Section 2.9.3 specifies the behavior if Downstream Port Containment has been triggered.

Comment [VS95]: ECN: Downstream Port containment (DPC)

- For a Port with DL_Down status, the Transaction Layer is not required to accept received TLPs from the Data Link Layer, provided that these TLPs have not been acknowledged by the Data Link Layer. Such TLPs do not modify receive Flow Control credits.

For a Downstream Port, DL_Down status is handled by:

- initializing back to their default state any buffers or internal states associated with outstanding requests transmitted Downstream
 - Note: Port configuration registers must not be affected, except as required to update status associated with the transition to DL_Down
- for Non-Posted Requests, forming completions for any Requests submitted by the device core for Transmission, returning Unsupported Request Completion Status, then discarding the Requests
 - This is a reported error associated with the Function for the (virtual) Bridge associated with the Port (see Section 6.2). For Root Ports, the reporting of this error is optional.
 - Non-Posted Requests already being processed by the Transaction Layer, for which it may not be practical to return Completions, are discarded

 Note: This is equivalent to the case where the Request had been Transmitted but not yet Completed before the Link status became DL_Down

- These cases are handled by the Requester using the Completion Timeout mechanism

Note: The point at which a Non-Posted Request becomes "uncompletable" is implementation specific.

- ❑ The Port must terminate any PME_Turn_Off handshake Requests targeting the Port in such a way that the Port is considered to have acknowledged the PME_Turn_Off request (see the Implementation Note in Section 5.3.3.2.1).
- ❑ The Port must handle Vendor Defined Message Requests as described in Section 2.2.8.6 (e.g., silently discard Vendor Defined Type 1 Messages Requests that it is not designed to receive) since the DL_Down prevents the Request from reaching its targeted Function.
- ❑ for all other Posted Requests, discarding the Requests
 - o This is a reported error associated with the Function for the (virtual) Bridge associated with the Port (see Section 6.2), and must be reported as an Unsupported Request. For Root Ports, the reporting of this error is optional.
 - o For a Posted Request already being processed by the Transaction Layer, the Port is permitted not to report the error.

 Note: This is equivalent to the case where the Request had been Transmitted before the Link status became DL_Down

 Note: The point at which a Posted Request becomes "unreportable" is implementation specific.
- ❑ discarding all Completions submitted by the device core for Transmission

For an Upstream Port, DL_Down status is handled as a reset by:

- ❑ returning all PCI Express-specific registers, state machines and externally observable state to the specified default or initial conditions (except for registers defined as sticky – see Section 7.4)
- ❑ discarding all TLPs being processed
- ❑ (for Switch and Bridge) propagating hot reset to all associated Downstream Ports. In Switches that support Link speeds greater than 5.0 GT/s, the Upstream Port must direct the LTSSM of each Downstream Port to the Hot Reset state, but not hold the LTSSMs in that state. This permits each Downstream Port to begin Link training immediately after its hot reset completes. This behavior is recommended for all Switches.

2.9.2. Transaction Layer Behavior in DL_Up Status

- ❑ DL_Up status indicates that a connection has been established with another component on the associated Link. This section specifies the Transaction Layer's behavior when the Data Link Layer reports entry to the DL_Up status to the Transaction Layer, indicating that the Link is operational. The Transaction layer of a Port with DL_Up Status must accept received TLPs that conform to the other rules of this specification.

For a Downstream Port on a Root Complex or a Switch:

- ❑ When transitioning from a non-DL_Up Status to a DL_Up Status and the Auto Slot Power Limit Disable bit is Cleared in the Slot Control Register, the Port must initiate the transmission

Comment [VS96]: ECN: Downstream Port containment (DPC)

of a Set_Slot_Power_Limit Message to the other component on the Link to convey the value programmed in the Slot Power Limit Scale and Value fields of the Slot Capabilities register. This Transmission is optional if the Slot Capabilities register has not yet been initialized.

2.9.3. Transaction Layer Behavior During Downstream Port Containment

Comment [VS97]: ECN: Enhanced DPC

During Downstream Port Containment (DPC), the LTSSM associated with the Downstream Port is directed to the Disabled state. Once it reaches the Disabled state, it remains there as long as the DPC Trigger Status bit in the DPC Status register is Set. See Section 6.2.10 for requirements on how long software must leave the Downstream Port in DPC. This section specifies the Transaction Layer's behavior once DPC has been triggered, and as long as the Downstream Port remains in DPC.

- Once DPC has been triggered, no additional (Upstream) TLPs are accepted from the Data Link Layer.
- If the condition that triggered DPC was associated with an Upstream TLP, any subsequent Upstream TLPs that were already accepted from the Data Link Layer must be discarded silently.

The Downstream Port handles (Downstream) TLPs submitted by the device core in the following manner.

- If the condition that triggered DPC was associated with a Downstream TLP, any prior Downstream TLPs are permitted to be dropped silently or transmitted before the Link goes down. Otherwise, the following rules apply.
- For each Non-Posted Request, the Port must return a Completion and discard the Request silently. The Completer ID field must contain the value associated with the Downstream Port.
 - If the DPC Completion Control bit is Set in the DPC Control register, then Completions are generated with Unsupported Request (UR) Completion Status.
 - If the DPC Completion Control bit is Clear, Completions are generated with Completer Abort (CA) Completion Status.
- The Port must terminate any PME_Turn_Off handshake Requests targeting the Port in such a way that the Port is considered to have acknowledged the PME_Turn_Off Request (see the Implementation Note in Section 5.3.3.2.1).
- The Port must handle Vendor Defined Message Requests as described in Section 2.2.8.6. (e.g., silently discard Vendor Defined Type 1 Message Requests that it is not designed to receive) since the DL_Down prevents the Request from reaching its targeted Function.
- For all other Posted Requests and Completions, the Port must silently discard the TLP.

For any outstanding Non-Posted Requests where DPC being triggered prevents their associated Completions from being returned, the following apply:

- For Root Ports that support RP Extensions for DPC, the Root Port may track certain Non-Posted Requests, and when DPC is triggered, synthesize a Completion for each tracked Request. This helps avoid Completion Timeouts that would otherwise occur as a side-effect of DPC being triggered. Each synthesized Completion must have a UR or CA Completion Status as

determined by the DPC Completion Control bit. The set of Non-Posted Requests that get tracked is implementation-specific, but it is strongly recommended that all Non-Posted Requests that are generated by host processor instructions (e.g., "read", "write", "load", "store", or one that corresponds to an AtomicOp) be tracked. Other candidates for tracking include peer-to-peer Requests coming from other Root Ports and Requests coming from Root Complex Integrated Endpoints.

- Otherwise, the associated Requesters may encounter Completion Timeouts. The software solution stack should comprehend and account for this possibility.

3

3. Data Link Layer Specification

The Data Link Layer acts as an intermediate stage between the Transaction Layer and the Physical Layer. Its primary responsibility is to provide a reliable mechanism for exchanging Transaction Layer Packets (TLPs) between the two components on a Link.

3.1. Data Link Layer Overview

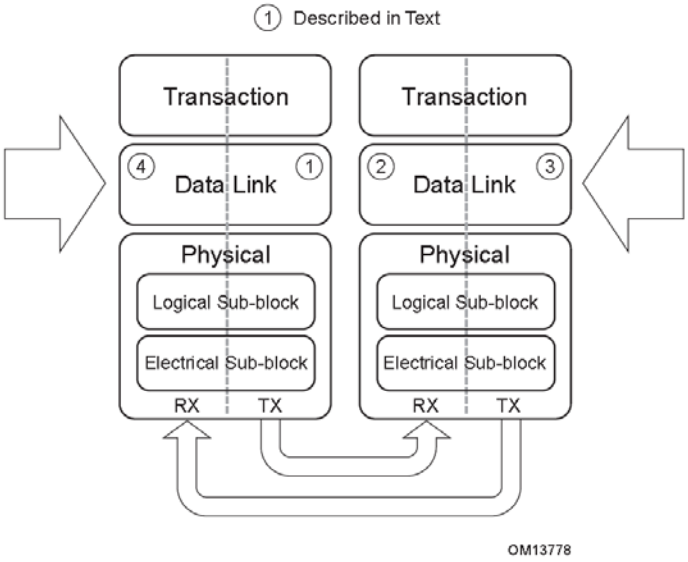


Figure 3-1: Layering Diagram Highlighting the Data Link Layer

The Data Link Layer is responsible for reliably conveying Transaction Layer Packets (TLPs) supplied by the Transaction Layer across a PCI Express Link to the other component's Transaction Layer. Services provided by the Data Link Layer include:

Data Exchange:

- Accept TLPs for transmission from the Transmit Transaction Layer and convey them to the Transmit Physical Layer
- Accept TLPs received over the Link from the Physical Layer and convey them to the Receive Transaction Layer

Error Detection and Retry:

- TLP Sequence Number and LCRC generation
- Transmitted TLP storage for Data Link Layer Retry
- Data integrity checking for TLPs and Data Link Layer Packets (DLLPs)
- Positive and negative acknowledgement DLLPs
- Error indications for error reporting and logging mechanisms
- Link Acknowledgement Timeout replay mechanism

Initialization and power management:

- Track Link state and convey active/reset/disconnected state to Transaction Layer

Data Link Layer Packets (DLLPs) are:

- used for Link Management functions including TLP acknowledgement, power management, and exchange of Flow Control information.
- transferred between Data Link Layers of the two directly connected components on a Link

DLLPs are sent point-to-point, between the two components on one Link. TLPs are routed from one component to another, potentially through one or more intermediate components.

Data Integrity checking for DLLPs and TLPs is done using a CRC included with each packet sent across the Link. DLLPs use a 16-bit CRC and TLPs (which can be much longer than DLLPs) use a 32-bit LCRC. TLPs additionally include a sequence number, which is used to detect cases where one or more entire TLPs have been lost.

Received DLLPs which fail the CRC check are discarded. The mechanisms which use DLLPs may suffer a performance penalty from this loss of information, but are self-repairing such that a successive DLLP will supersede any information lost.

TLPs which fail the data integrity checks (LCRC and sequence number), or which are lost in transmission from one component to another, are re-sent by the Transmitter. The Transmitter stores a copy of all TLPs sent, re-sending these copies when required, and purges the copies only when it receives a positive acknowledgement of error-free receipt from the other component. If a positive acknowledgement has not been received within a specified time period, the Transmitter will automatically start re-transmission. The Receiver can request an immediate re-transmission using a negative acknowledgement.

The Data Link Layer appears as an information conduit with varying latency to the Transaction Layer. On any given individual Link all TLPs fed into the Transmit Data Link Layer (1 and 3) will appear at the output of the Receive Data Link Layer (2 and 4) in the same order at a later time, as

illustrated in Figure 3-1. The latency will depend on a number of factors, including pipeline latencies, width and operational frequency of the Link, transmission of electrical signals across the Link, and delays caused by Data Link Layer Retry. As a result of these delays, the Transmit Data Link Layer (1 and 3) can apply backpressure to the Transmit Transaction Layer, and the Receive Data Link Layer (2 and 4) communicates the presence or absence of valid information to the Receive Transaction Layer.

3.2. Data Link Control and Management State Machine

The Data Link Layer tracks the state of the Link. It communicates Link status with the Transaction and Physical Layers, and performs Link management through the Physical Layer. The Data Link Layer contains the Data Link Control and Management State Machine (DLCMSM) to perform these tasks. The states for this machine are described below, and are shown in Figure 3-2.

States:

- DL_Inactive – Physical Layer reporting Link is non-operational or nothing is connected to the Port
- DL_Init – Physical Layer reporting Link is operational, initialize Flow Control for the default Virtual Channel
- DL_Active – Normal operation mode

Status Outputs:

- DL_Down – The Data Link Layer is not communicating with the component on the other side of the Link.
- DL_Up – The Data Link Layer is communicating with the component on the other side of the Link.

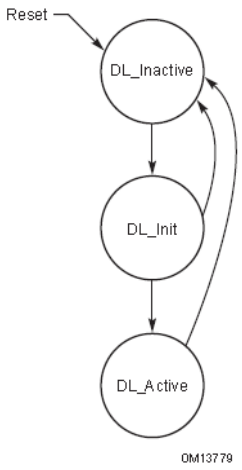


Figure 3-2: Data Link Control and Management State Machine

3.2.1. Data Link Control and Management State Machine Rules

Rules per state:

- DL_Inactive
 - Initial state following PCI Express hot, warm, or cold reset (see Section 6.6). Note that DL states are unaffected by an FLR (see Section 6.6).
 - Upon entry to DL_Inactive
 - Reset all Data Link Layer state information to default values
 - Discard the contents of the Data Link Layer Retry Buffer (see Section 3.5)
 - While in DL_Inactive:
 - Report DL_Down status to the Transaction Layer as well as to the rest of the Data Link Layer

 Note: This will cause the Transaction Layer to discard any outstanding transactions and to terminate internally any attempts to transmit a TLP. For a Downstream Port, this is equivalent to a "Hot-Remove." For an Upstream Port, having the Link go down is equivalent to a hot reset (see Section 2.9).
 - Discard TLP information from the Transaction and Physical Layers
 - Do not generate or accept DLLPs
 - Exit to DL_Init if:
 - Indication from the Transaction Layer that the Link is not disabled by software and the Physical Layer reports Physical LinkUp = 1b

- DL_Init
 - While in DL_Init:
 - Initialize Flow Control for the default Virtual Channel, VC0, following the Flow Control initialization protocol described in Section 3.3
 - Report DL_Down status while in state FC_INIT1; DL_Up status in state FC_INIT2
 - The Data Link Layer of a Port with DL_Down status is permitted to discard any received TLPs provided that it does not acknowledge those TLPs by sending one or more Ack DLLPs
 - Exit to DL_Active if:
 - Flow Control initialization completes successfully, and the Physical Layer continues to report Physical LinkUp = 1b
 - Terminate attempt to initialize Flow Control for VC0 and Exit to DL_Inactive if:
 - Physical Layer reports Physical LinkUp = 0b
- DL_Active
 - DL_Active is referred to as the normal operating state
 - While in DL_Active:
 - Accept and transfer TLP information with the Transaction and Physical Layers as specified in this chapter
 - Generate and accept DLLPs as specified in this chapter
 - Report DL_Up status to the Transaction and Data Link Layers
 - Exit to DL_Inactive if:
 - Physical Layer reports Physical LinkUp = 0b
 - ~~Upstream components are optionally permitted to~~ Downstream Ports that are Surprise Down Error Reporting Capable (see Section 7.8.6) must treat this transition from DL_Active to DL_Inactive as a Surprise Down error, except in the following cases where this error detection is blocked:
 - If the Secondary Bus Reset in Bridge Control register has been set to 1b by software, then the subsequent transition to DL_Inactive must not be considered an error.
 - If the Link Disable bit has been set to 1b by software, then the subsequent transition to DL_Inactive must not be considered an error.
 - If a Switch Downstream Port transitions to DL_Inactive due to an event above that Port, that transition to DL_Inactive must not be considered an error. Example events include the Switch Upstream Port propagating Hot Reset, the Switch Upstream Link transitioning to DL_Down, and the Secondary Bus Reset bit in the Switch Upstream Port being set to 1b.

Comment [VS98]: B16

- If a PME_Turn_Off Message has been sent through this Port, then the subsequent transition to DL_Inactive must not be considered an error.
- Note that the DL_Inactive transition for this condition will not occur until a power off, a reset, or a request to restore the Link is sent to the Physical layer.
- Note also that in the case where the PME_Turn_Off/PME_TO_Ack handshake fails to complete successfully, a Surprise Down error may be detected.
- If the Port is associated with a hot-pluggable slot (Hot-Plug Capable bit in the Slot Capabilities register set to 1b), and the Hot-Plug Surprise bit in the Slot Capabilities register is set to 1b, then any transition to DL_Inactive must not be considered an error.
- If the Port is associated with a hot-pluggable slot (Hot-Plug Capable bit in the Slot Capabilities register set to 1b), and Power Controller Control bit in Slot Control register is 1b(Off), then any transition to DL Inactive must not be considered an error.

Error blocking initiated by one or more of the above cases must remain in effect until the Port exits DL_Active and subsequently returns to DL_Active with none of the blocking cases in effect at the time of the return to DL_Active.

Note that the transition out of DL_Active is simply the expected transition as anticipated per the error detection blocking condition.

If implemented, this is a reported error associated with the detecting Port (see Section 6.2).



IMPLEMENTATION NOTE

Physical Layer Throttling

Note that there are conditions where the Physical Layer may be temporarily unable to accept TLPs and DLLPs from the Data Link Layer. The Data Link Layer must comprehend this by providing mechanisms for the Physical Layer to communicate this condition, and for TLPs and DLLPs to be temporarily blocked by the condition.

3.3. Flow Control Initialization Protocol

Before starting normal operation following power-up or interconnect Reset, it is necessary to initialize Flow Control for the default Virtual Channel, VC0 (see Section 6.6). In addition, when additional Virtual Channels (VCs) are enabled, the Flow Control initialization process must be completed for each newly enabled VC before it can be used (see Section 2.6.1). This section describes the initialization process that is used for all VCs. Note that since VC0 is enabled before all other VCs, no TLP traffic of any kind will be active prior to initialization of VC0. However, when additional VCs are being initialized there will typically be TLP traffic flowing on other, already enabled, VCs. Such traffic has no direct effect on the initialization process for the additional VC(s).

There are two states in the VC initialization process. These states are:

- FC_INIT1
- FC_INIT2

The rules for this process are given in the following section.

3.3.1. Flow Control Initialization State Machine Rules

- If at any time during initialization for VCs 1-7 the VC is disabled, the flow control initialization process for the VC is terminated
- Rules for state FC_INIT1:
 - Entered when initialization of a VC (VCx) is required
 - Entrance to DL_Init state (VCx = VC0)
 - When a VC (VCx = VC1-7) is enabled by software (see Sections 7.11 and 7.18)
 - While in FC_INIT1:
 - Transaction Layer must block transmission of TLPs using VCx
 - Transmit the following three InitFC1 DLLPs for VCx in the following relative order:
 - InitFC1 – P (first)
 - InitFC1 – NP (second)
 - InitFC1 – Cpl (third)
 - The three InitFC1 DLLPs must be transmitted at least once every 34 µs.
 - Time spent in the Recovery or Configuration LTSSM states does not contribute to this limit.
 - It is strongly encouraged that the InitFC1 DLLP transmissions are repeated frequently, particularly when there are no other TLPs or DLLPs available for transmission.
 - Except as needed to ensure at least the required frequency of InitFC1 DLLP transmission, the Data Link Layer must not block other transmissions
 - Note that this includes all Physical Layer initiated transmissions (for example, Ordered Sets), Ack and Nak DLLPs (when applicable), and TLPs using VCs that have previously completed initialization (when applicable)
 - Process received InitFC1 and InitFC2 DLLPs:
 - Record the indicated FC unit values
 - Set Flag FI1 once FC unit values have been recorded for each of P, NP, and Cpl for VCx

- Exit to FC_INIT2 if:
 - Flag FI1 has been set indicating that FC unit values have been recorded for each of P, NP, and Cpl for VCx

- Rules for state FC_INIT2:
 - While in FC_INIT2:
 - Transaction Layer must block transmission of TLPs using VCx
 - Transmit the following three InitFC2 DLLPs for VCx in the following relative order:
 - InitFC2 – P (first)
 - InitFC2 – NP (second)
 - InitFC2 – Cpl (third)
 - The three InitFC2 DLLPs must be transmitted at least once every 34 µs.
 - Time spent in the Recovery or Configuration LTSSM states does not contribute to this limit.
 - It is strongly encouraged that the InitFC2 DLLP transmissions are repeated frequently, particularly when there are no other TLPs or DLLPs available for transmission.
 - Except as needed to ensure at least the required frequency of InitFC2 DLLP transmission, the Data Link Layer must not block other transmissions
 - Note that this includes all Physical Layer initiated transmissions (for example, Ordered Sets), Ack and Nak DLLPs (when applicable), and TLPs using VCs that have previously completed initialization (when applicable)
 - Process received InitFC1 and InitFC2 DLLPs:
 - Ignore the indicated FC unit values
 - Set flag FI2 on receipt of any InitFC2 DLLP for VCx
 - Set flag FI2 on receipt of any TLP on VCx, or any UpdateFC DLLP for VCx
 - Signal completion and exit if:
 - Flag FI2 has been set

IMPLEMENTATION NOTE

Example of Flow Control Initialization

Figure 3-3 illustrates an example of the Flow Control initialization protocol for VC0 between a Switch and a Downstream component. In this example, each component advertises the minimum permitted values for each type of Flow Control credit. For both components the largest Max_Payload_Size value supported is 1024 bytes, corresponding to a data payload credit advertisement of 040h. All DLLPs are shown as received without error.

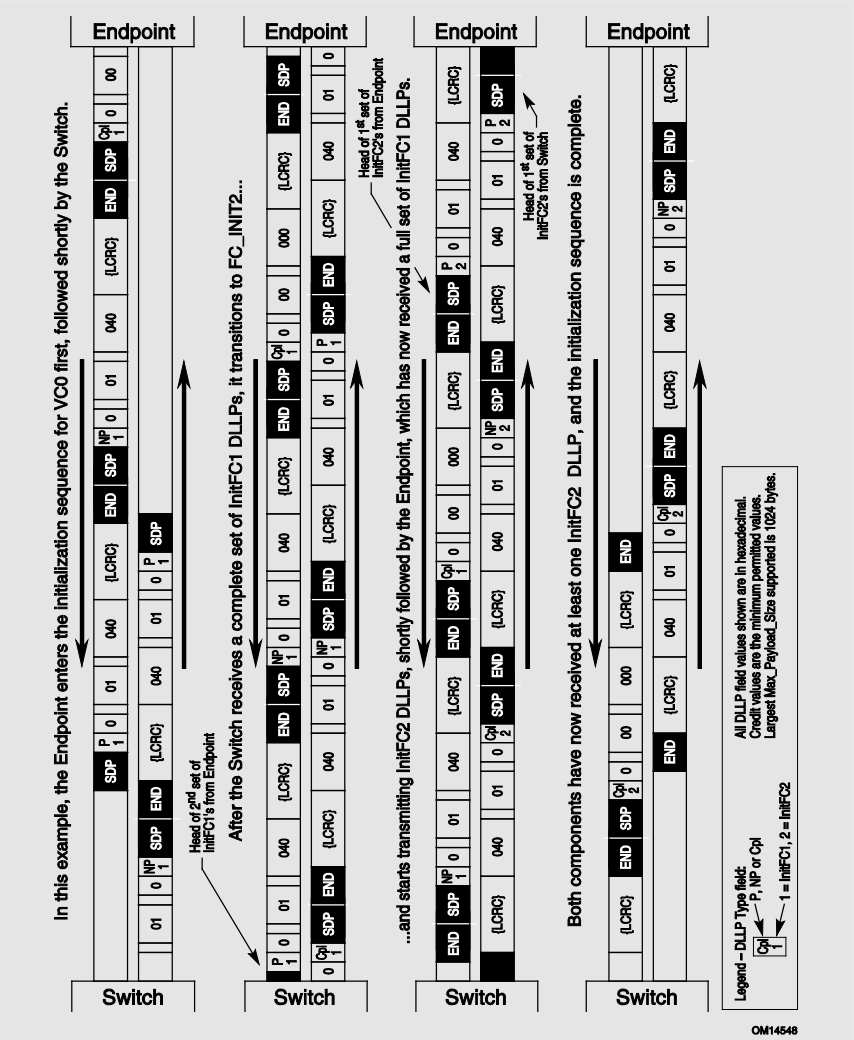


Figure 3-3: VC0 Flow Control Initialization Example with 8b/10b Encoding-based Framing

3.4. Data Link Layer Packets (DLLPs)

The following DLLPs are used to support Link operations:

- Ack DLLP: TLP Sequence number acknowledgement; used to indicate successful receipt of some number of TLPs
- Nak DLLP: TLP Sequence number negative acknowledgement; used to initiate a Data Link Layer Retry
- InitFC1, InitFC2, and UpdateFC DLLPs: For Flow Control
- DLLPs used for Power Management

3.4.1. Data Link Layer Packet Rules

All DLLP fields marked Reserved (sometimes abbreviated as R) must be filled with all 0's when a DLLP is formed. Values in such fields must be ignored by Receivers. The handling of Reserved values in encoded fields is specified for each case.

All DLLPs include the following fields:

- DLLP Type - Specifies the type of DLLP. The defined encodings are shown in Table 3-1.
- 16-bit CRC

See Figure 3-4 below.

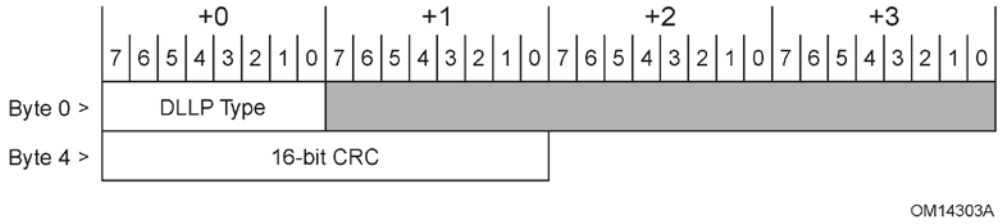


Figure 3-4: DLLP Type and CRC Fields

Table 3-1: DLLP Type Encodings

Encodings	DLLP Type
0000 0000	Ack
0001 0000	Nak
0010 0000	PM_Enter_L1
0010 0001	PM_Enter_L23
0010 0011	PM_Active_State_Request_L1
0010 0100	PM_Request_Ack
0011 0000	Vendor Specific – Not used in normal operation
0100 $0v_2v_1v_0$	InitFC1-P (v[2:0] specifies Virtual Channel)
0101 $0v_2v_1v_0$	InitFC1-NP
0110 $0v_2v_1v_0$	InitFC1-Cpl
1100 $0v_2v_1v_0$	InitFC2-P
1101 $0v_2v_1v_0$	InitFC2-NP
1110 $0v_2v_1v_0$	InitFC2-Cpl
1000 $0v_2v_1v_0$	UpdateFC-P
1001 $0v_2v_1v_0$	UpdateFC-NP
1010 $0v_2v_1v_0$	UpdateFC-Cpl
All other encodings	Reserved

- For Ack and Nak DLLPs (see Figure 3-5):
 - The AckNak_Seq_Num field is used to indicate what TLPs are affected
 - Transmission and Reception is handled by the Data Link Layer according to the rules provided in Section 3.5.
- For InitFC1, InitFC2, and UpdateFC DLLPs:
 - The HdrFC field contains the credit value for headers of the indicated type (P, NP, or Cpl)
 - The DataFC field contains the credit value for payload Data of the indicated type (P, NP, or Cpl)
 - The packet formats are shown in Figure 3-6, Figure 3-7, and Figure 3-8
 - Transmission is triggered by the Data Link Layer when initializing Flow Control for a Virtual Channel (see Section 3.3), and following Flow Control initialization by the Transaction Layer according to the rules in Section 2.6
 - Checked for integrity on reception by the Data Link Layer and if correct, the information content of the DLLP is passed to the Transaction Layer. If the check fails, the information is discarded.

 Note: InitFC1 and InitFC2 DLLPs are used only for VC initialization

- Power Management (PM) DLLPs (see Figure 3-9):
 - Transmission is triggered by the component's power management logic according to the rules in Chapter 5
 - Checked for integrity on reception by the Data Link Layer, then passed to the component's power management logic
- Vendor Specific (see Figure 3-10)

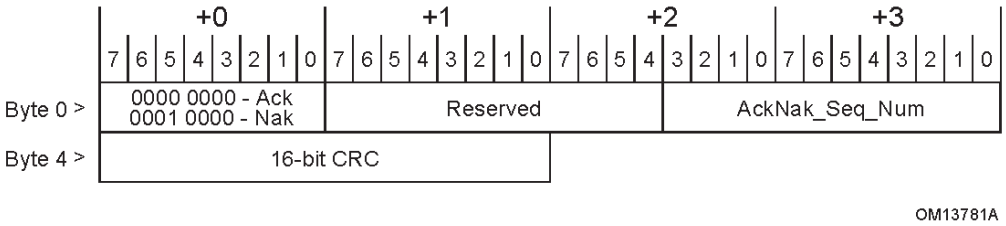


Figure 3-5: Data Link Layer Packet Format for Ack and Nak

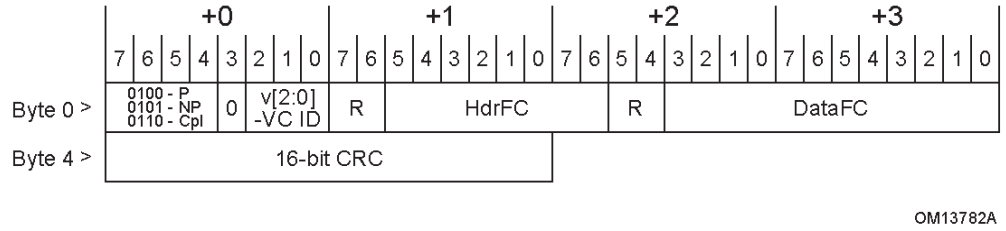


Figure 3-6: Data Link Layer Packet Format for InitFC1

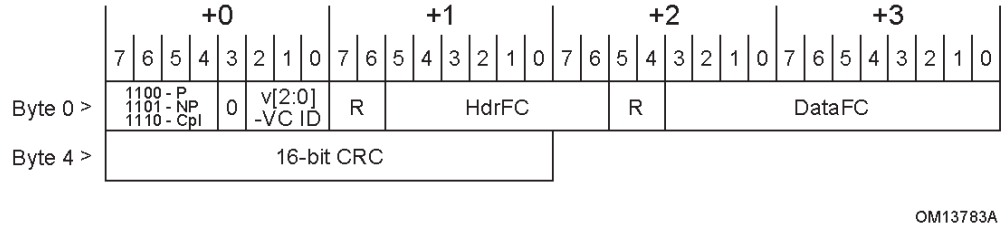


Figure 3-7: Data Link Layer Packet Format for InitFC2

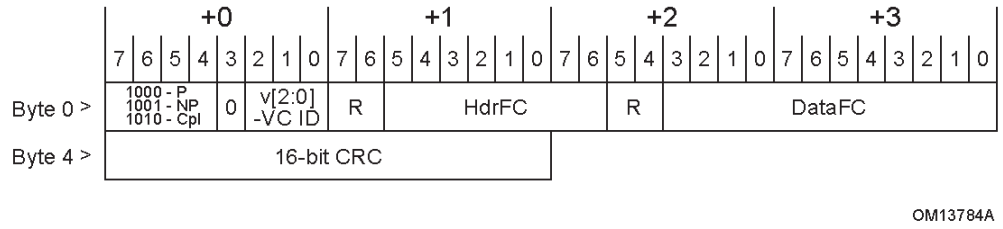


Figure 3-8: Data Link Layer Packet Format for UpdateFC

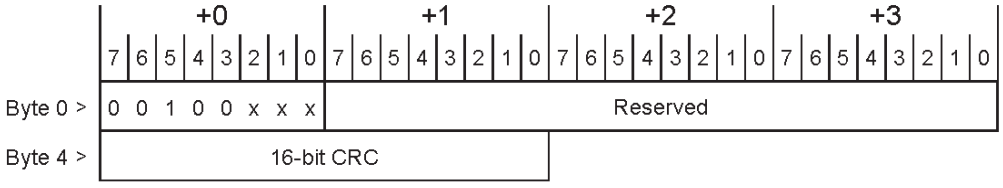


Figure 3-9: PM Data Link Layer Packet Format

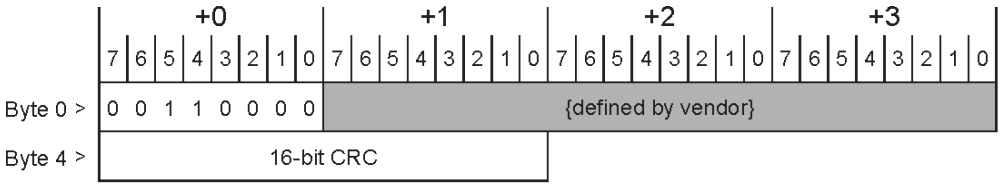


Figure 3-10: Vendor Specific Data Link Layer Packet Format

The following are the characteristics and rules associated with Data Link Layer Packets (DLLPs):

- DLLPs are differentiated from TLPs when they are presented to, or received from, the Physical Layer.
- DLLP data integrity is protected using a 16-bit CRC
- The CRC value is calculated using the following rules (see Figure 3-11):
 - The polynomial used for CRC calculation has a coefficient expressed as 100Bh
 - The seed value (initial value for CRC storage registers) is FFFFh
 - CRC calculation starts with bit 0 of byte 0 and proceeds from bit 0 to bit 7 of each byte
 - Note that CRC calculation uses all bits of the DLLP, regardless of field type, including Reserved fields. The result of the calculation is complemented, then placed into the 16-bit CRC field of the DLLP as shown in Table 3-2.

Table 3-2: Mapping of Bits into CRC Field

CRC Result Bit	Corresponding Bit Position in the 16-Bit CRC Field
0	7
1	6
2	5
3	4
4	3
5	2
6	1
7	0
8	15
9	14
10	13
11	12
12	11
13	10
14	9
15	8

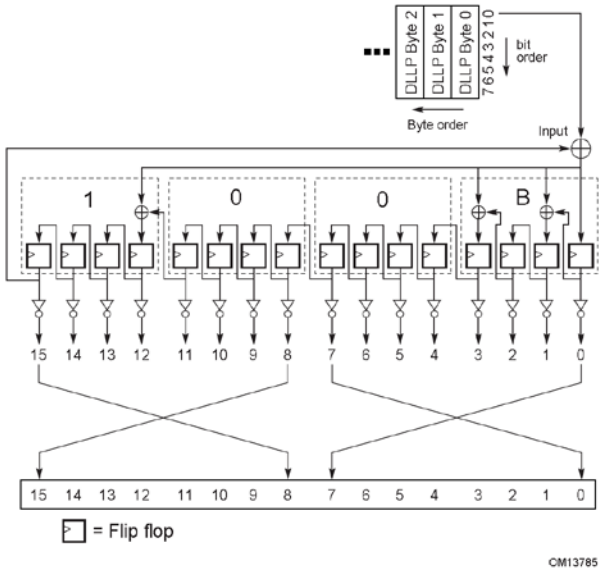


Figure 3-11: Diagram of CRC Calculation for DLLPs

3.5. Data Integrity

3.5.1. Introduction

The Transaction Layer provides TLP boundary information to the Data Link Layer. This allows the Data Link Layer to apply a TLP Sequence Number and a Link CRC (LCRC) for error detection to the TLP. The Receive Data Link Layer validates received TLPs by checking the TLP Sequence Number, LCRC code and any error indications from the Receive Physical Layer. In case any of these errors are in a TLP, Data Link Layer Retry is used for recovery.

The format of a TLP with the TLP Sequence Number and LCRC code applied is shown in Figure 3-12.

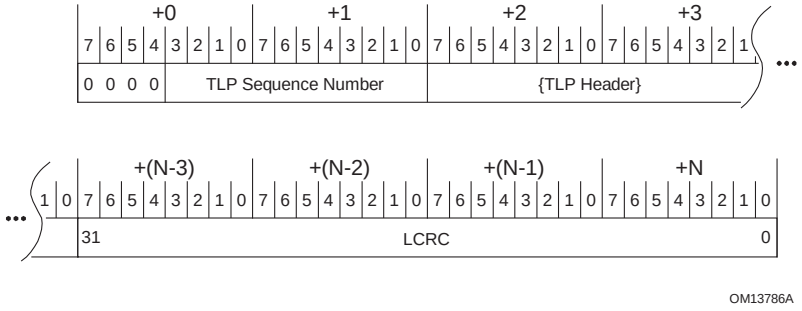


Figure 3-12: TLP with LCRC and TLP Sequence Number Applied

On Ports that support Protocol Multiplexing, packets containing a non-zero value in Symbol +0, bits 7:4 are PMUX Packets. For TLPs, these bits must be 0000b. See Appendix G for details.

On Ports that do not support Protocol Multiplexing, Symbol +0, bits 7:4 are Reserved.

3.5.2. LCRC, Sequence Number, and Retry Management (TLP Transmitter)

The TLP transmission path through the Data Link Layer (paths labeled 1 and 3 in Figure 3-1) prepares each TLP for transmission by applying a sequence number, then calculating and appending a Link CRC (LCRC) which is used to ensure the integrity of TLPs during transmission across a Link from one component to another. TLPs are stored in a retry buffer, and are re-sent unless a positive acknowledgement of receipt is received from the other component. If repeated attempts to transmit a TLP are unsuccessful, the Transmitter will determine that the Link is not operating correctly, and instruct the Physical Layer to retrain the Link (via the LTSSM Recovery state, Section 4.2.6). If Link retraining fails, the Physical Layer will indicate that the Link is no longer up, causing the DLCMSM to move to the DL_Inactive state.

The mechanisms used to determine the TLP LCRC and the Sequence Number and to support Data Link Layer Retry are described in terms of conceptual "counters" and "flags." This description does not imply nor require a particular implementation and is used only to clarify the requirements.

3.5.2.1. LCRC and Sequence Number Rules (TLP Transmitter)

The following counters and timer are used to explain the remaining rules in this section:

- The following 12-bit counters are used:

 NEXT_TRANSMIT_SEQ – Stores the packet sequence number applied to TLPs

 - Set to 000h in DL_Inactive state

 - ACKD_SEQ – Stores the sequence number acknowledged in the most recently received Ack or Nak DLLP.

 - Set to FFFh in DL_Inactive state

- The following 2-bit counter is used:

 REPLAY_NUM – Counts the number of times the Retry Buffer has been re-transmitted

 - Set to 00b in DL_Inactive state

- The following timer is used:

 - REPLAY_TIMER - Counts time that determines when a replay is required, according to the following rules:

 - Started at the last Symbol of any TLP transmission or retransmission, if not already running
 - For each replay, reset and restart REPLAY_TIMER when sending the last Symbol of the first TLP to be retransmitted
 - Resets and restarts for each Ack DLLP received while there are more unacknowledged TLPs outstanding, if, and only if, the received Ack DLLP acknowledges some TLP in the retry buffer
 - Note: This ensures that REPLAY_TIMER is reset only when forward progress is being made
 - Reset and hold until restart conditions are met for each Nak received (except during a replay) or when the REPLAY_TIMER expires
 - Not advanced during Link retraining (holds its value when the LTSSM is in the Recovery or Configuration state). Refer to Sections 4.2.5.3 and 4.2.5.4.
 - If Protocol Multiplexing is supported, optionally not advanced during the reception of PMUX Packets (see Appendix G).
 - Resets and holds when there are no outstanding unacknowledged TLPs

The following rules describe how a TLP is prepared for transmission before being passed to the Physical Layer:

- The Transaction Layer indicates the start and end of the TLP to the Data Link Layer while transferring the TLP
 - The Data Link Layer treats the TLP as a "black box" and does not process or modify the contents of the TLP
- Each TLP is assigned a 12-bit sequence number when it is accepted from the Transmit side of Transaction Layer
 - Upon acceptance of the TLP from the Transaction Layer, the packet sequence number is applied to the TLP by:
 - prepending the 12-bit value in NEXT_TRANSMIT_SEQ to the TLP
 - prepending 4 Reserved bits to the TLP, preceding the sequence number (see Figure 3-13)
 - If the equation:

 (NEXT_TRANSMIT_SEQ – ACKD_SEQ) mod 4096 >= 2048

 is true, the Transmitter must cease accepting TLPs from the Transaction Layer until the equation is no longer true
 - Following the application of NEXT_TRANSMIT_SEQ to a TLP accepted from the Transmit side of the Transaction Layer, NEXT_TRANSMIT_SEQ is incremented (except in the case where the TLP is nullified):

 NEXT_TRANSMIT_SEQ := (NEXT_TRANSMIT_SEQ + 1) mod 4096

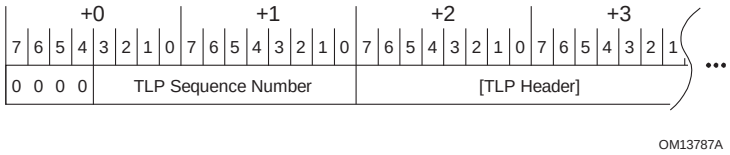


Figure 3-13: TLP Following Application of TLP Sequence Number and Reserved Bits

- TLP data integrity is protected during transfer between Data Link Layers using a 32-bit LCRC
- The LCRC value is calculated using the following mechanism (Figure 3-14):
 - The polynomial used has coefficients expressed as 04C1 1DB7h
 - The seed value (initial value for LCRC storage registers) is FFFF FFFFh
 - The LCRC is calculated using the TLP following sequence number application (see Figure 3-13)
 - LCRC calculation starts with bit 0 of byte 0 (bit 8 of the TLP sequence number) and proceeds from bit 0 to bit 7 of each successive byte.
 - Note that LCRC calculation uses all bits of the TLP, regardless of field type, including Reserved fields

- The remainder of the LCRC calculation is complemented, and the complemented result bits are mapped into the 32-bit LCRC field as shown in Table 3-3.

Table 3-3: Mapping of Bits into LCRC Field

LCRC Result Bit	Corresponding Bit Position in the 32-Bit LCRC Field
0	7
1	6
2	5
3	4
4	3
5	2
6	1
7	0
8	15
9	14
10	13
11	12
12	11
13	10
14	9
15	8
16	23
17	22
18	21
19	20
20	19
21	18
22	17
23	16
24	31
25	30
26	29
27	28
28	27
29	26
30	25
31	24

The 32-bit LCRC field is appended to the TLP following the bytes received from the Transaction Layer (see Figure 3-12).

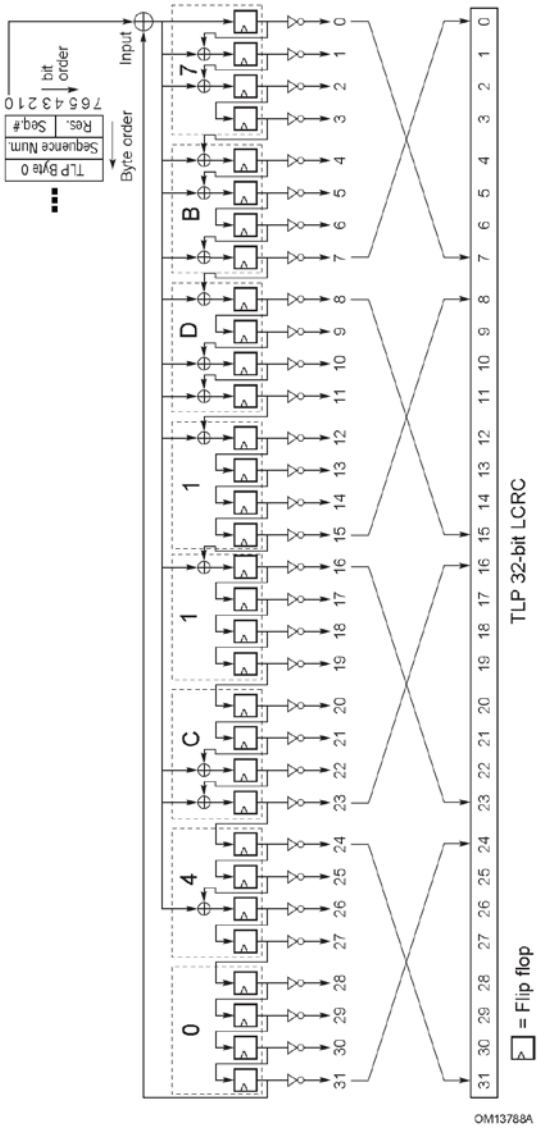


Figure 3-14: Calculation of LCRC

To support cut-through routing of TLPs, a Transmitter is permitted to modify a transmitted TLP to indicate that the Receiver must ignore that TLP ("nullify" the TLP).

- A Transmitter is permitted to nullify a TLP being transmitted; to do this in a way which will robustly prevent misinterpretation or corruption, the Transmitter must do the following:
 - transmit all DWs of the TLP when the Physical Layer is using 128b/130b encoding (see Section 4.2.2.3.1)
 - use the remainder of the calculated LCRC value without inversion (the logical inverse of the value normally used)
 - indicate to the Transmit Physical Layer that the TLP is nullified
- When this is done, the Transmitter does not increment NEXT_TRANSMIT_SEQ

The following rules describe the operation of the Data Link Layer Retry Buffer, from which TLPs are re-transmitted when necessary:

- Copies of Transmitted TLPs must be stored in the Data Link Layer Retry Buffer, except for nullified TLPs.

When a replay is initiated, either due to reception of a Nak or due to REPLAY_TIMER expiration, the following rules describe the sequence of operations that must be followed:

- If all TLPs transmitted have been acknowledged (the Retry Buffer is empty), terminate replay, otherwise continue.
- Increment REPLAY_NUM. When the replay is initiated by the reception of a Nak which acknowledged some TLPs in the retry buffer, REPLAY_NUM is reset. It is then permitted (but not required) to be incremented.
 - If REPLAY_NUM rolls over from 11b to 00b, the Transmitter signals the Physical Layer to retrain the Link, and waits for the completion of retraining before proceeding with the replay. This is a reported error associated with the Port (see Section 6.2).

 Note that Data Link Layer state, including the contents of the Retry Buffer, are not reset by this action unless the Physical Layer reports Physical LinkUp = 0b (causing the Data Link Control and Management State Machine to transition to the DL_Inactive state).
 - If REPLAY_NUM does not roll over from 11b to 00b, continue with the replay.
- Block acceptance of new TLPs from the Transmit Transaction Layer.
- Complete transmission of any TLP currently being transmitted.
- Retransmit unacknowledged TLPs, starting with the oldest unacknowledged TLP and continuing in original transmission order
 - Reset and restart REPLAY_TIMER when sending the last Symbol of the first TLP to be retransmitted
 - Once all unacknowledged TLPs have been re-transmitted, return to normal operation.
 - If any Ack or Nak DLLPs are received during a replay, the Transmitter is permitted to complete the replay without regard to the Ack or Nak DLLP(s), or to skip retransmission of any newly acknowledged TLPs.

- Once the Transmitter has started to resend a TLP, it must complete transmission of that TLP in all cases.

- Ack and Nak DLLPs received during a replay must be processed, and may be collapsed
 - Example: If multiple Acks are received, only the one specifying the latest Sequence Number value must be considered – Acks specifying earlier Sequence Number values are effectively "collapsed" into this one
 - Example: During a replay, Nak is received, followed by an Ack specifying a later Sequence Number – the Ack supersedes the Nak, and the Nak is ignored

 Note: Since all entries in the Retry Buffer have already been allocated space in the Receiver by the Transmitter's Flow Control gating logic, no further flow control synchronization is necessary.

- Re-enable acceptance of new TLPs from the Transmit Transaction Layer.

A replay can be initiated by the expiration of REPLAY_TIMER, or by the receipt of a Nak. The following rule covers the expiration of REPLAY_TIMER:

- If the Transmit Retry Buffer contains TLPs for which no Ack or Nak DLLP has been received, and (as indicated by REPLAY_TIMER) no Ack or Nak DLLP has been received for a period exceeding the time indicated in Table 3-4 for 2.5 GT/s mode operation, Table 3-5 for 5.0 GT/s mode operation, and Table 3-6 for 8.0 GT/s mode operation, the Transmitter initiates a replay.

 This is a Replay Timer Timeout error and it is a reported error associated with the Port (see Section 6.2).

The following formula defines the timeout count values for the REPLAY_TIMER. There are three tables: one that applies when operating in 2.5 GT/s mode, one that applies when operating in 5.0 GT/s mode, and one that applies when operating in 8.0 GT/s mode. The values are specified according to the largest TLP payload size and Link width.

The values are calculated using the formula (note – this is simply three times the Ack Latency value – see Section 3.5.3.1):

$$\left(\frac{(Max_Payload_Size + TLPOverhead) * AckFactor}{LinkWidth} + InternalDelay\right) * 3$$

where

Max_Payload_Size	is the value in the Max_Payload_Size field of the Device Control register. For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. For a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the common Max_Payload_Size setting must be used. For a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the selected Max_Payload_Size setting is implementation specific, but it is recommended to use the largest Max_Payload_Size setting across all Functions.
TLPOverhead	represents the additional TLP components which consume Link bandwidth (TLP Prefix, header, LCRC, framing Symbols) and is treated here as a constant value of 28 Symbols.
AckFactor	represents the number of maximum size TLPs which can be received before an Ack is sent, and is used to balance Link bandwidth efficiency and retry buffer size – the value varies according to Max_Payload_Size and Link width, and is defined in Table 3-7, Table 3-8, and Table 3-6.
LinkWidth	is the operating width of the Link.
InternalDelay	represents the internal processing delays for received TLPs and transmitted DLLPs, and is treated here as a constant value of 19 Symbol Times for 2.5 GT/s mode operation, 70 Symbol Times for 5.0 GT/s mode operation, and 115 Symbol Times for 8.0 GT/s mode operation. For M-PCIe implementations HS-G1 shall use 19 Symbol Times for InternalDelay.

Comment [VS99]: Mpcie_28August2014 Errata C6

TLP Transmitters and compliance tests must base replay timing as measured at the Port of the TLP Transmitter. Timing starts with either the last Symbol of a transmitted TLP, or else the last Symbol of a received Ack DLLP, whichever determines the oldest unacknowledged TLP. Timing ends with the First Symbol of TLP retransmission.

It is strongly recommended that a TLP Transmitter not perform a TLP retransmission due to Ack delay if the delay is potentially caused by the Ack's Link needing to exit L0s before it can transmit the Ack. This might be accomplished by statically adjusting the REPLAY_TIMER to allow for the

L0s exit latency of the Ack's Link, or by sensing if the Ack's Link is in the process of exiting L0s when the REPLAY_TIMER expires.

When measuring replay timing to the point when TLP retransmission begins, compliance tests must allow for any other TLP or DLLP transmission already in progress in that direction (thus preventing the TLP retransmission). Also, compliance tests must allow for implementations that statically adjust the REPLAY_TIMER by the L0s exit latency of the Ack's Link. Finally, if the retransmitted TLP's Link is enabled for L0s, compliance tests must allow for its L0s exit latency.[38]

Table 3-4: Unadjusted[39] REPLAY_TIMER Limits for 2.5 GT/s Mode Operation[40] by Link Width and Max_Payload_Size (Symbol Times) Tolerance: -0%/+100%

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	711	384	219	201	174	144	99
	256	1248	651	354	321	270	216	135
	512	1677	867	462	258	327	258	156
	1024	3213	1635	846	450	582	450	252
	2048	6285	3171	1614	834	1095	834	444
	4096	12429	6243	3150	1602	2118	1602	828

Table 3-5: Unadjusted[39] REPLAY_TIMER Limits for 5.0 GT/s Mode Operation by Link Width and Max_Payload_Size (Symbol Times) Tolerance: -0%/+100%

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	864	537	372	354	327	297	252
	256	1401	804	507	474	423	369	288
	512	1830	1020	615	411	480	411	309
	1024	3366	1788	999	603	735	603	405
	2048	6438	3324	1767	987	1248	987	597
	4096	12582	6396	3303	1755	2271	1755	981

Comment [VS100]: Mpcie_28August2014 Errata C6 [MC6]

[38] Note that L0s exit latency is affected by the value of the Extended Synch bit in the Link Control register.

[39] A TLP Transmitter is permitted to adjust its REPLAY_TIMER to allow for L0s exit latency as described in the text preceding the table.

[40] Note that for M-PCIe implementations, the values in Table 3-4 also apply for HS-G1.

Table 3-6: Unadjusted[39] REPLAY_TIMER Limits for 8.0 GT/s Operation by Link Width and Max_Payload_Size (Symbol Times) Tolerance: -0%/+100%

		Link Operating Width						
		x1	x2	x4	x8	x12	x16	x32
Max_Payload_Size (bytes)	128	999	672	507	489	462	432	387
	256	1536	939	642	609	558	504	423
	512	1965	1155	750	546	615	546	444
	1024	3501	1923	1134	738	870	738	540
	2048	6573	3459	1902	1122	1383	1122	732
	4096	12717	6531	3438	1890	2406	1890	1116



IMPLEMENTATION NOTE

Recommended Priority of Scheduled Transmissions

When multiple DLLPs of the same type are scheduled for transmission but have not yet been transmitted, it is possible in many cases to "collapse" them into a single DLLP. For example, if a scheduled Ack DLLP transmission is stalled waiting for another transmission to complete, and during this time another Ack is scheduled for transmission, it is only necessary to transmit the second Ack, since the information it provides will supersede the information in the first Ack.

In addition to any TLP from the Transaction Layer (or the Retry Buffer, if a replay is in progress), Multiple DLLPs of different types may be scheduled for transmission at the same time, and must be prioritized for transmission. The following list shows the preferred priority order for selecting information for transmission. Note that the priority of the vendor specific DLLP is not listed, as this is completely implementation specific, and there is no recommended priority. Note that this priority order is a guideline, and that in all cases a fairness mechanism is highly recommended to ensure that no type of traffic is blocked for an extended or indefinite period of time by any other type of traffic. Note that the Ack Latency value and REPLAY_TIMER limit specify requirements measured at the Port of the component, and the internal arbitration policy of the component must ensure that these externally measured requirements are met.

1) Completion of any transmission (TLP or DLLP) currently in progress (highest priority)
2) Nak DLLP transmissions
3) Ack DLLP transmissions scheduled for transmission as soon as possible due to:
 receipt of a duplicate TLP –OR–
 expiration of the Ack latency timer (see Section 3.5.3.1)
4) FC DLLP transmissions required to satisfy Section 2.6
5) Retry Buffer re-transmissions
6) TLPs from the Transaction Layer
7) FC DLLP transmissions other than those required to satisfy Section 2.6
8) All other DLLP transmissions (lowest priority)

3.5.2.2. Handling of Received DLLPs

Since Ack/Nak and Flow Control DLLPs affect TLPs flowing in the opposite direction across the Link, the TLP transmission mechanisms in the Data Link Layer are also responsible for Ack/Nak and Flow Control DLLPs received from the other component on the Link. These DLLPs are processed according to the following rules (see Figure 3-15):

- If the Physical Layer indicates a Receiver Error, discard any DLLP currently being received and free any storage allocated for the DLLP. Note that reporting such errors to software is done by the Physical Layer (and, therefore, are not reported by the Data Link Layer).
- For all received DLLPs, the CRC value is checked by:
 - applying the same algorithm used for calculation of transmitted DLLPs to the received DLLP, not including the 16-bit CRC field of the received DLLP
 - comparing the calculated result with the value in the CRC field of the received DLLP
 - if not equal, the DLLP is corrupt
 - A corrupt received DLLP is discarded, and is a reported error associated with the Port (see Section 6.2).
- A received DLLP which is not corrupt, but which uses unsupported DLLP Type encodings is discarded without further action. This is not considered an error.
- Non-zero values in Reserved fields are ignored.
- Receivers must process all DLLPs received at the rate they are received

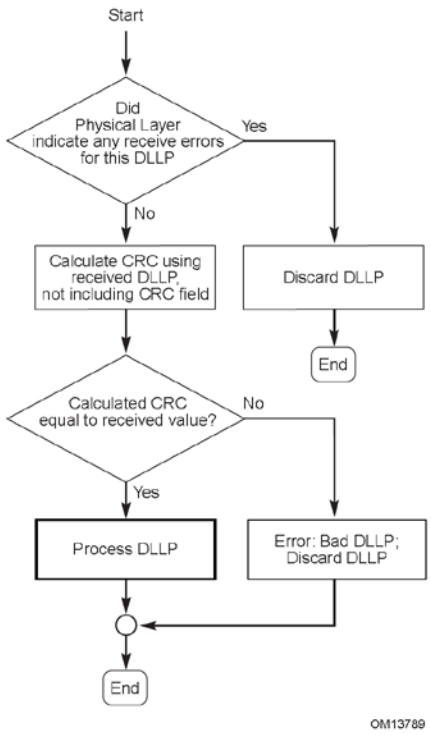


Figure 3-15: Received DLLP Error Check Flowchart

- Received FC DLLPs are passed to the Transaction Layer
- Received PM DLLPs are passed to the component's power management control logic
- For Ack and Nak DLLPs, the following steps are followed (see Figure 3-16):
 - If the Sequence Number specified by the AckNak_Seq_Num does not correspond to an unacknowledged TLP, or to the value in ACKD_SEQ, the DLLP is discarded
 - This is a Data Link Protocol Error which is a reported error associated with the Port (see Section 6.2).

 Note that it is not an error to receive an Ack DLLP when there are no outstanding unacknowledged TLPs, including the time between reset and the first TLP transmission, as long as the specified Sequence Number matches the value in ACKD_SEQ.
 - If the AckNak_Seq_Num does not specify the Sequence Number of the most recently acknowledged TLP, then the DLLP acknowledges some TLPs in the retry buffer:
 - Purge from the retry buffer all TLPs from the oldest to the one corresponding to the AckNak_Seq_Num

- Load ACKD_SEQ with the value in the AckNak_Seq_Num field
- Reset REPLAY_NUM and REPLAY_TIMER

o If the DLLP is a Nak, initiate a replay (see above)

Note: Receipt of a Nak is not a reported error.

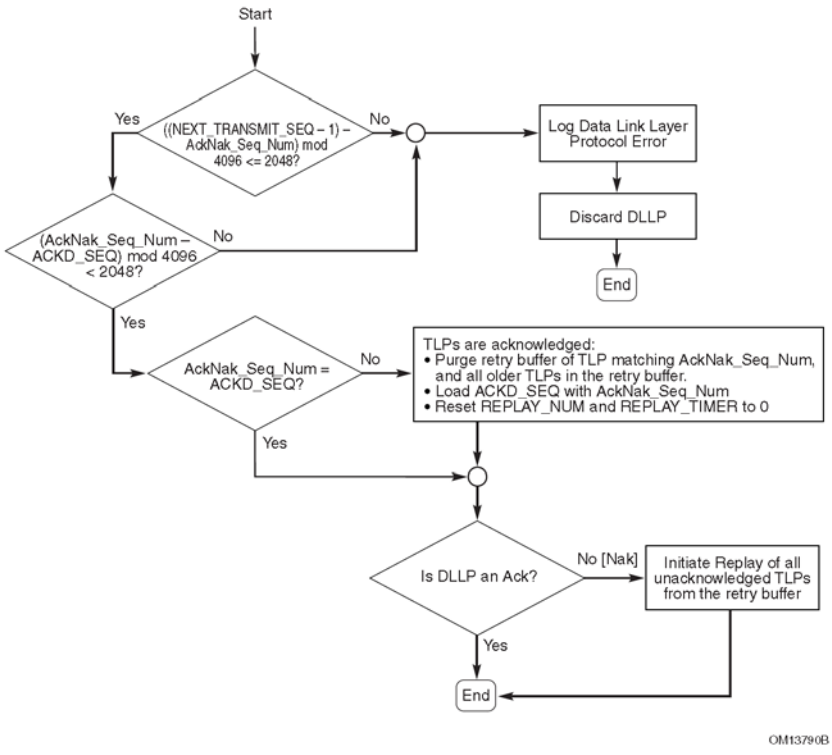


Figure 3-16: Ack/Nak DLLP Processing Flowchart

The following rules describe the operation of the Data Link Layer Retry Buffer, from which TLPs are re-transmitted when necessary:

❑ Copies of Transmitted TLPs must be stored in the Data Link Layer Retry Buffer

3.5.3. LCRC and Sequence Number (TLP Receiver)

The TLP Receive path through the Data Link Layer (paths labeled 2 and 4 in Figure 3-1) processes TLPs received by the Physical Layer by checking the LCRC and sequence number, passing the TLP to the Receive Transaction Layer if OK and requesting a replay if corrupted.

The mechanisms used to check the TLP LCRC and the Sequence Number and to support Data Link Layer Retry are described in terms of conceptual "counters" and "flags." This description does not imply or require a particular implementation and is used only to clarify the requirements.

3.5.3.1. LCRC and Sequence Number Rules (TLP Receiver)

The following counter, flag, and timer are used to explain the remaining rules in this section:

- The following 12-bit counter is used:
 - NEXT_RCV_SEQ – Stores the expected Sequence Number for the next TLP
 - Set to 000h in DL_Inactive state
- The following flag is used:
 - NAK_SCHEDULED
 - Cleared when in DL_Inactive state
- The following timer is used:
 - AckNak_LATENCY_TIMER – Counts time that determines when an Ack DLLP becomes scheduled for transmission, according to the following rules:
 - Set to 0 in DL_Inactive state
 - Restart from 0 each time an Ack or Nak DLLP is scheduled for transmission; Reset to 0 when all TLPs received have been acknowledged with an Ack DLLP
 - If there are initially no unacknowledged TLPs and a TLP is then received, the AckNak_LATENCY_TIMER starts counting only when the TLP has been forwarded to the Receive Transaction Layer

The following rules are applied in sequence to describe how received TLPs are processed, and what events trigger the transmission of Ack and Nak DLLPs (see Figure 3-17):

- If the Physical Layer indicates a Receiver Error, discard any TLP currently being received and free any storage allocated for the TLP. Note that reporting such errors to software is done by the Physical Layer (and so are not reported by the Data Link Layer).
 - If a TLP was being received at the time the Receiver Error was indicated and the NAK_SCHEDULED flag is clear,
 - schedule a Nak DLLP for transmission immediately
 - set the NAK_SCHEDULED flag
- If the Physical Layer reports that the received TLP was nullified, and the LCRC is the logical NOT of the calculated value, discard the TLP and free any storage allocated for the TLP. This is not considered an error.
- If TLP was nullified but the LCRC does not match the logical NOT of the calculated value, the TLP is corrupt - discard the TLP and free any storage allocated for the TLP.
 - If the NAK_SCHEDULED flag is clear,
 - schedule a Nak DLLP for transmission immediately
 - set the NAK_SCHEDULED flag

 This is a Bad TLP error and is a reported error associated with the Port (see Section 6.2).

- The LCRC value is checked by:
 - applying the same algorithm used for calculation (above) to the received TLP, not including the 32-bit LCRC field of the received TLP
 - comparing the calculated result with the value in the LCRC field of the received TLP
 - if not equal, the TLP is corrupt - discard the TLP and free any storage allocated for the TLP
 - If the NAK_SCHEDULED flag is clear,
 - schedule a Nak DLLP for transmission immediately
 - set the NAK_SCHEDULED flag

 This is a Bad TLP error and is a reported error associated with the Port (see Section 6.2).
- If the TLP Sequence Number is not equal to the expected value, stored in NEXT_RCV_SEQ:
 - discard the TLP and free any storage allocated for the TLP
 - If the TLP Sequence Number satisfies the following equation:

 (NEXT_RCV_SEQ - TLP Sequence Number) mod 4096 <= 2048

 the TLP is a duplicate, and an Ack DLLP is scheduled for transmission (per transmission priority rules)
 - Otherwise, the TLP is out of sequence (indicating one or more lost TLPs):
 - if the NAK_SCHEDULED flag is clear,
 - schedule a Nak DLLP for transmission immediately
 - set the NAK_SCHEDULED flag

 This is a Bad TLP error and is a reported error associated with the Port (see Section 6.2).

 Regardless of the state of the NAK_SCHEDULED flag, it is permitted for this to be a reported error associated with the Port (see Section 6.2), and this permitted behavior is illustrated in Figure 3-17. However, in order to prevent error pollution it is recommended that the Port only report such an error when the NAK_SCHEDULED flag is clear.

- If the TLP Sequence Number is equal to the expected value stored in NEXT_RCV_SEQ:
 - The four Reserved bits, TLP Sequence Number, and LCRC (see Figure 3-12) are removed and the remainder of the TLP is forwarded to the Receive Transaction Layer
 - The Data Link Layer indicates the start and end of the TLP to the Transaction Layer while transferring the TLP
 - The Data Link Layer treats the TLP as a "black box" and does not process or modify the contents of the TLP
 - Note that the Receiver Flow Control mechanisms do not account for any received TLPs until the TLP(s) are forwarded to the Receive Transaction Layer
 - NEXT_RCV_SEQ is incremented
 - If Set, the NAK_SCHEDULED flag is cleared

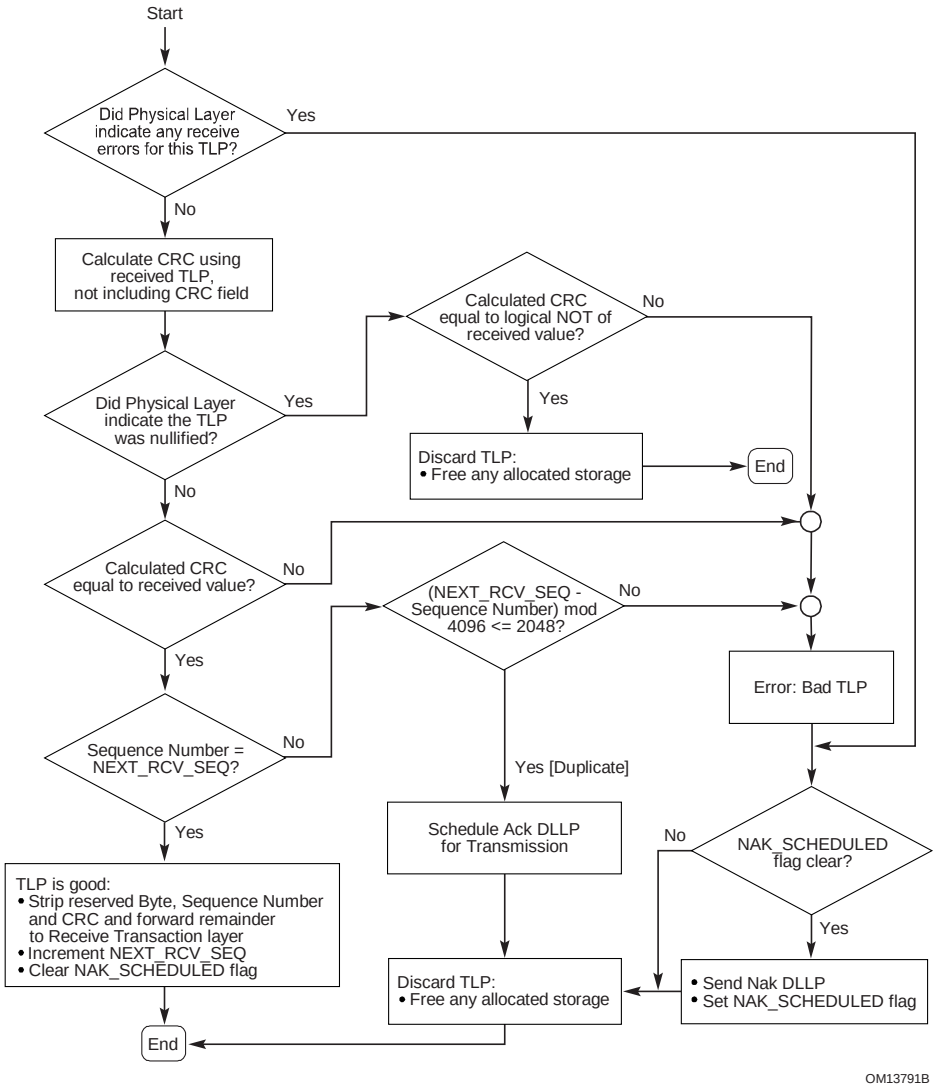


Figure 3-17: Receive Data Link Layer Handling of TLPs

- A TLP Receiver must schedule an Ack DLLP such that it will be transmitted no later than when all of the following conditions are true:
 - The Data Link Control and Management State Machine is in the DL_Active state
 - TLPs have been forwarded to the Receive Transaction Layer, but not yet acknowledged by sending an Ack DLLP
 - The AckNak_LATENCY_TIMER reaches or exceeds the value specified in Table 3-7 for 2.5 GT/s mode operation, Table 3-8 for 5.0 GT/s mode operation, and Table 3-9 for 8.0 GT/s mode operation
 - The Link used for Ack DLLP transmission is already in L0 or has transitioned to L0
 - Note: if not already in L0, the Link must transition to L0 in order to transmit the Ack DLLP
 - Another TLP or DLLP is not currently being transmitted on the Link used for Ack DLLP transmission
 - The NAK_SCHEDULED flag is clear
 - Note: The AckNak_LATENCY_TIMER must be restarted from 0 each time an Ack or Nak DLLP is scheduled for transmission
- Data Link Layer Ack DLLPs may be scheduled for transmission more frequently than required
- Data Link Layer Ack and Nak DLLPs specify the value (NEXT_RCV_SEQ - 1) in the AckNak_Seq_Num field

Table 3-7, Table 3-8, and Table 3-9 define the threshold values for the AckNak_LATENCY_TIMER, which for any specific case is called the Ack Latency. The values are calculated using the formula:

$$\frac{(Max_Payload_Size + TLPOverhead) * AckFactor}{LinkWidth} + InternalDelay$$

where

Max_Payload_Size	is the value in the Max_Payload_Size field of the Device Control register. For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. For a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the common Max_Payload_Size setting must be used. For a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the selected Max_Payload_Size setting is implementation specific, but it is recommended to use the smallest Max_Payload_Size setting across all Functions.
TLPOverhead	represents the additional TLP components which consume Link bandwidth (TLP Prefix, header, LCRC, framing Symbols) and is treated here as a constant value of 28 Symbols.

AckFactor	represents the number of maximum size TLPs which can be received before an Ack is sent, and is used to balance Link bandwidth efficiency and retry buffer size – the value varies according to Max_Payload_Size and Link width, and is defined in Table 3-7, Table 3-8, and Table 3-9.
LinkWidth	is the operating width of the Link.
InternalDelay	represents the internal processing delays for received TLPs and transmitted DLLPs, and is treated here as a constant value of 19 Symbol Times for 2.5 GT/s mode operation, 70 Symbol Times for 5.0 GT/s operation, and 115 Symbol Times for 8.0 GT/s operation. For M-PCIe implementations HS-G1 shall use 19 Symbol Times for InternalDelay.

Comment [VS101]: Mpcie_28August2014 Errata C6

TLP Receivers and compliance tests must base Ack Latency timing as measured at the Port of the TLP Receiver, starting with the time the last Symbol of a TLP is received to the first Symbol of the Ack DLLP being transmitted.

When measuring until the Ack DLLP is transmitted, compliance tests must allow for any TLP or other DLLP transmission already in progress in that direction (thus preventing the Ack DLLP transmission). If L0s is enabled, compliance tests must allow for the L0s exit latency of the Link in the direction that the Ack DLLP is being transmitted. If the Extended Synch bit of the Link Control register is Set, compliance tests must also allow for its effect on L0s exit latency.

TLP Receivers are not required to adjust their Ack DLLP scheduling based upon L0s exit latency or the value of the Extended Synch bit.

Table 3-7: Ack Transmission Latency Limit and AckFactor for 2.5 GT/s Mode Operation[41] by Link Width and Max Payload (Symbol Times)

Comment [VS102]: Mpcie_28August2014 Errata C6 [MC6]

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	237 AF = 1.4	128 AF = 1.4	73 AF = 1.4	67 AF = 2.5	58 AF = 3.0	48 AF = 3.0	33 AF = 3.0
	256	416 AF = 1.4	217 AF = 1.4	118 AF = 1.4	107 AF = 2.5	90 AF = 3.0	72 AF = 3.0	45 AF = 3.0
	512	559 AF = 1.0	289 AF = 1.0	154 AF = 1.0	86 AF = 1.0	109 AF = 2.0	86 AF = 2.0	52 AF = 2.0
	1024	1071 AF = 1.0	545 AF = 1.0	282 AF = 1.0	150 AF = 1.0	194 AF = 2.0	150 AF = 2.0	84 AF = 2.0
	2048	2095 AF = 1.0	1057 AF = 1.0	538 AF = 1.0	278 AF = 1.0	365 AF = 2.0	278 AF = 2.0	148 AF = 2.0
	4096	4143 AF = 1.0	2081 AF = 1.0	1050 AF = 1.0	534 AF = 1.0	706 AF = 2.0	534 AF = 2.0	276 AF = 2.0

[41] Note that for M-PCIe implementations, the values in Table 3 7 also apply for HS-G1.

Table 3-8: Ack Transmission Latency Limit and AckFactor for 5.0 GT/s Mode Operation by Link Width and Max Payload (Symbol Times)

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	288 AF = 1.4	179 AF = 1.4	124 AF = 1.4	118 AF = 2.5	109 AF = 3.0	99 AF = 3.0	84 AF = 3.0
	256	467 AF = 1.4	268 AF = 1.4	169 AF = 1.4	158 AF = 2.5	141 AF = 3.0	123 AF = 3.0	96 AF = 3.0
	512	610 AF = 1.0	340 AF = 1.0	205 AF = 1.0	137 AF = 1.0	160 AF = 2.0	137 AF = 2.0	103 AF = 2.0
	1024	1122 AF = 1.0	596 AF = 1.0	333 AF = 1.0	201 AF = 1.0	245 AF = 2.0	201 AF = 2.0	135 AF = 2.0
	2048	2146 AF = 1.0	1108 AF = 1.0	589 AF = 1.0	329 AF = 1.0	416 AF = 2.0	329 AF = 2.0	199 AF = 2.0
	4096	4194 AF = 1.0	2132 AF = 1.0	1101 AF = 1.0	585 AF = 1.0	757 AF = 2.0	585 AF = 2.0	327 AF = 2.0

Table 3-9: Ack Transmission Latency Limit and AckFactor for 8.0 GT/s Mode Operation by Link Width and Max Payload (Symbol Times)

		Link Operating Width						
		x1	**x2**	**x4**	**x8**	**x12**	**x16**	**x32**
Max_Payload_Size (bytes)	**128**	333 AF = 1.4	224 AF = 1.4	169 AF = 1.4	163 AF = 2.5	154 AF = 3.0	144 AF = 3.0	129 AF = 3.0
	256	512 AF = 1.4	313 AF = 1.4	214 AF = 1.4	203 AF = 2.5	186 AF = 3.0	168 AF = 3.0	141 AF = 3.0
	512	655 AF = 1.0	385 AF = 1.0	250 AF = 1.0	182 AF = 1.0	205 AF = 2.0	182 AF = 2.0	148 AF = 2.0
	1024	1167 AF = 1.0	641 AF = 1.0	378 AF = 1.0	246 AF = 1.0	290 AF = 2.0	246 AF = 2.0	180 AF = 2.0
	2048	2191 AF = 1.0	1153 AF = 1.0	634 AF = 1.0	374 AF = 1.0	461 AF = 2.0	374 AF = 2.0	244 AF = 2.0
	4096	4239 AF = 1.0	2177 AF = 1.0	1146 AF = 1.0	630 AF = 1.0	802 AF = 2.0	630 AF = 2.0	372 AF = 2.0

IMPLEMENTATION NOTE

Retry Buffer Sizing

The Retry Buffer should be large enough to ensure that under normal operating conditions, transmission is never throttled because the retry buffer is full. In determining the optimal buffer size, one must consider the Ack Latency value, Ack delay caused by the Receiver already transmitting another TLP, the delays caused by the physical Link interconnect, and the time required to process the received Ack DLLP.

Given two components A and B, the L0s exit latency required by A's Receiver should be accounted for when sizing A's transmit retry buffer, as is demonstrated in the following example:

- A exits L0s on its Transmit path to B and starts transmitting a long burst of write Requests to B
- B initiates L0s exit on its Transmit path to A, but the L0s exit time required by A's Receiver is large
- Meanwhile, B is unable to send Ack DLLPs to A, and A stalls due to lack of Retry Buffer space
- The Transmit path from B to A returns to L0, B transmits an Ack DLLP to A, and the stall is resolved

This stall can be avoided by matching the size of a component's Transmitter Retry Buffer to the L0s exit latency required by the component's Receiver, or, conversely, by matching the Receiver L0s exit latency to the desired size of the Retry Buffer.

AckFactor values were chosen to allow implementations to achieve good performance without requiring an uneconomically large retry buffer. To enable consistent performance across a general purpose interconnect with differing implementations and applications, it is necessary to set the same requirements for all components without regard to the application space of any specific component. If a component does not require the full transmission bandwidth of the Link, it may reduce the size of its retry buffer below the minimum size required to maintain available retry buffer space with the Ack Latency values specified.

Note that the Ack Latency values specified ensure that the range of permitted outstanding TLP Sequence Numbers will never be the limiting factor causing transmission stalls.

4

4. Physical Layer Specification

4.1. Introduction

The Physical Layer isolates the Transaction and Data Link Layers from the signaling technology used for Link data interchange. The Physical Layer is divided into the logical and electrical sub-blocks (see Figure 4-1).

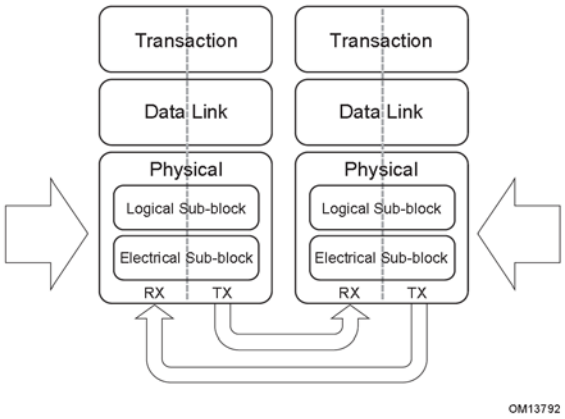


Figure 4-1: Layering Diagram Highlighting Physical Layer

4.2. Logical Sub-block

The logical sub-block has two main sections: a Transmit section that prepares outgoing information passed from the Data Link Layer for transmission by the electrical sub-block, and a Receiver section that identifies and prepares received information before passing it to the Data Link Layer.

The logical sub-block and electrical sub-block coordinate the state of each Transceiver through a status and control register interface or functional equivalent. The logical sub-block directs control and management functions of the Physical Layer.

PCI Express uses 8b/10b encoding when the data rate is 2.5 GT/s or 5.0 GT/s. For data rates greater than or equal to 8.0 GT/s, it uses a per-lane code along with physical layer encapsulation.

4.2.1. Encoding for 2.5 GT/s and 5.0 GT/s Data Rates

4.2.1.1. Symbol Encoding

PCI Express uses an 8b/10b transmission code. The definition of this transmission code is identical to that specified in ANSI X3.230-1994, clause 11 (and also IEEE 802.3z, 36.2.4). Using this scheme, 8-bit data characters are treated as 3 bits and 5 bits mapped onto a 4-bit code group and a 6-bit code group, respectively. The control bit in conjunction with the data character is used to identify when to encode one of the 12 Special Symbols included in the 8b/10b transmission code. These code groups are concatenated to form a 10-bit Symbol. As shown in Figure 4-2, ABCDE maps to abcdei and FGH maps to fghj.

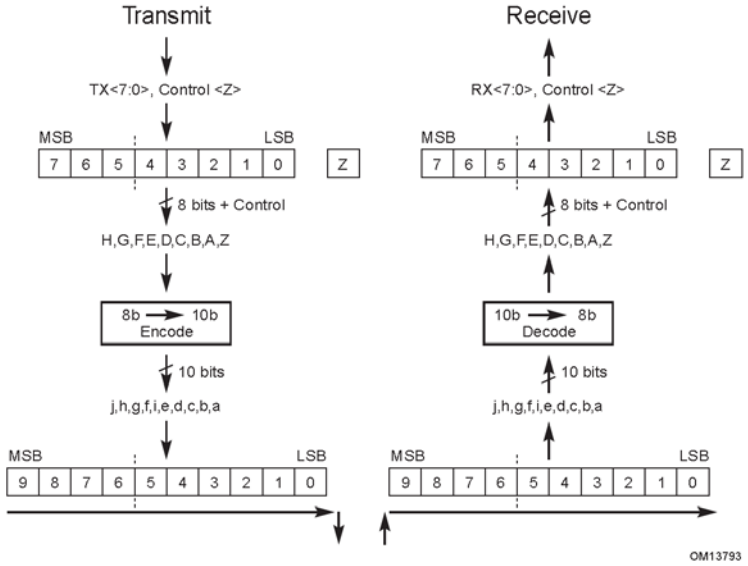


Figure 4-2: Character to Symbol Mapping

4.2.1.1.1. Serialization and De-serialization of Data

The bits of a Symbol are placed on a Lane starting with bit "a" and ending with bit "j." Examples are shown in Figure 4-3 and Figure 4-4.

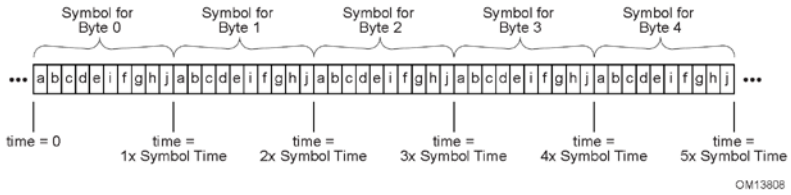


Figure 4-3: Bit Transmission Order on Physical Lanes - x1 Example

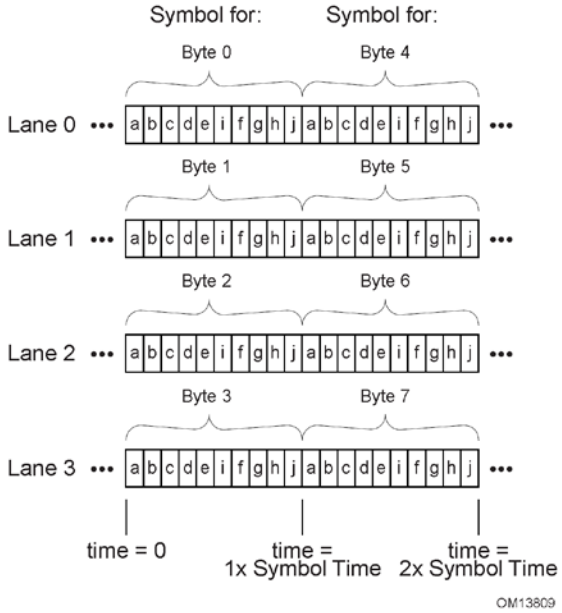


Figure 4-4: Bit Transmission Order on Physical Lanes - x4 Example

4.2.1.1.2. Special Symbols for Framing and Link Management (K Codes)

The 8b/10b encoding scheme provides Special Symbols that are distinct from the Data Symbols used to represent Characters. These Special Symbols are used for various Link Management mechanisms described later in this chapter. Special Symbols are also used to frame DLLPs and TLPs, using distinct Special Symbols to allow these two types of Packets to be quickly and easily distinguished.

Table 4-1 shows the Special Symbols used for PCI Express and provides a brief description for each. These Symbols will be discussed in greater detail in following sections. Each of these Special Symbols, as well as the data Symbols, must be interpreted by looking at the 10-bit Symbol in its entirety.

Table 4-1: Special Symbols

Encoding	Symbol	Name	Description
K28.5	COM	Comma	Used for Lane and Link initialization and management
K27.7	STP	Start TLP	Marks the start of a Transaction Layer Packet
K28.2	SDP	Start DLLP	Marks the start of a Data Link Layer Packet
K29.7	END	End	Marks the end of a Transaction Layer Packet or a Data Link Layer Packet
K30.7	EDB	EnD Bad	Marks the end of a nullified TLP
K23.7	PAD	Pad	Used in Framing and Link Width and Lane ordering negotiations
K28.0	SKP	Skip	Used for compensating for different bit rates for two communicating Ports
K28.1	FTS	Fast Training Sequence	Used within an Ordered Set to exit from L0s to L0
K28.3	IDL	Idle	Used in the Electrical Idle Ordered Set (EIOS)
K28.4			Reserved
K28.6			Reserved
K28.7	EIE	Electrical Idle Exit	Reserved in 2.5 GT/s Used in the Electrical Idle Exit Ordered Set (EIEOS) and sent prior to sending FTS at data rates other than 2.5 GT/s

4.2.1.1.3. 8b/10b Decode Rules

The Symbol tables for the valid 8b/10b codes are given in Appendix B. These tables have one column for the positive disparity and one column for the negative disparity.

A Transmitter is permitted to pick any disparity, unless otherwise required, when first transmitting differential data after being in an Electrical Idle state. The Transmitter must then follow proper 8b/10b encoding rules until the next Electrical Idle state is entered.

The initial disparity for a Receiver that detects an exit from Electrical Idle is set to the disparity of the first Symbol used to obtain Symbol lock. Disparity may also be-reinitialized if Symbol lock is lost and regained during the transmission of differential information due to an implementation specific number of errors. All following received Symbols after the initial disparity is set must be found in the proper column corresponding to the current running disparity.

If a received Symbol is found in the column corresponding to the incorrect running disparity or if the Symbol does not correspond to either column, the Physical Layer must notify the Data Link

Layer that the received Symbol is invalid. This is a Receiver Error, and is a reported error associated with the Port (see Section 6.2).

4.2.1.2. Framing and Application of Symbols to Lanes

There are two classes of framing and application of Symbols to Lanes. The first class consists of the Ordered Sets and the second class consists of TLPs and DLLPs. Ordered Sets are always transmitted serially on each Lane, such that a full Ordered Set appears simultaneously on all Lanes of a multi-Lane Link.

The Framing mechanism uses Special Symbol K28.2 "SDP" to start a DLLP and Special Symbol K27.7 "STP" to start a TLP. The Special Symbol K29.7 "END" is used to mark the end of either a TLP or a DLLP.

The conceptual stream of Symbols must be mapped from its internal representation, which is implementation dependent, onto the external Lanes. The Symbols are mapped onto the Lanes such that the first Symbol (representing Character 0) is placed onto Lane 0; the second is placed onto Lane 1; etc. The x1 Link represents a degenerate case and the mapping is trivial, with all Symbols placed onto the single Lane in order.

When no packet information or special Ordered Sets are being transmitted, the Transmitter is in the Logical Idle state. During this time idle data must be transmitted. The idle data must consist of the data byte 0 (00 Hexadecimal), scrambled according to the rules of Section 4.2.1.3 and 8b/10b encoded according to the rules of Section 4.2.1.1, in the same way that TLP and DLLP Data Symbols are scrambled and encoded. Likewise, when the Receiver is not receiving any packet information or special Ordered Sets, the Receiver is in Logical Idle and shall receive idle data as described above. During transmission of the idle data, the SKP Ordered Set must continue to be transmitted as specified in Section 4.2.7.

For the following rules, "placed" is defined to mean a requirement on the Transmitter to put the Symbol into the proper Lane of a Link.

- TLPs must be framed by placing an STP Symbol at the start of the TLP and an END Symbol or EDB Symbol at the end of the TLP (see Figure 4-5).
- DLLPs must be framed by placing an SDP Symbol at the start of the DLLP and an END Symbol at the end of the DLLP (see Figure 4-6).
- Logical Idle is defined to be a period of one or more Symbol Times when no information: TLPs, DLLPs or any type of Special Symbol is being Transmitted/Received. Unlike Electrical Idle, during Logical Idle the Idle Symbol (00h) is being transmitted and received.
 - When the Transmitter is in Logical Idle, the Logical Idle data (00h) shall be transmitted on all Lanes. This is scrambled according to the rules in Section 4.2.1.3.
 - Receivers must ignore incoming Logical Idle data, and must not have any dependency other than scramble sequencing on any specific data patterns.
- For Links wider than x1, the STP Symbol (representing the start of a TLP) must be placed in Lane 0 when starting Transmission of a TLP from a Logical Idle Link condition.
- For Links wider than x1, the SDP Symbol (representing the start of a DLLP) must be placed in Lane 0 when starting Transmission of a DLLP from a Logical Idle Link condition.

- The STP Symbol must not be placed on the Link more frequently than once per Symbol Time.
- The SDP Symbol must not be placed on the Link more frequently than once per Symbol Time.
- As long as the above rules are satisfied, TLP and DLLP Transmissions are permitted to follow each other successively.
- One STP Symbol and one SDP Symbol may be placed on the Link in the same Symbol Time.
 - Links wider than x4 can have STP and SDP Symbols placed in Lane 4*N, where N is a positive integer. For example, for x8, STP and SDP Symbols can be placed in Lanes 0 and 4; and for x16, STP and SDP Symbols can be placed in Lanes 0, 4, 8, or 12.
- For xN Links where N is 8 or more, if an END or EDB Symbol is placed in a Lane K, where K does not equal N-1, and is not followed by a STP or SDP Symbol in Lane K+1 (i.e., there is no TLP or DLLP immediately following), then PAD Symbols must be placed in Lanes K+1 to Lane N-1.
 - For example, on a x8 Link, if END or EDB is placed in Lane 3, PAD must be placed in Lanes 4 to 7, when not followed by STP or SDP.
- The EDB Symbol is used to mark the end of a nullified TLP. Refer to Section 3.5.2.1 for information on the usage of EDB.
- Receivers may optionally check for violations of the rules of this section. These checks are independently optional (see Section 6.2.3.4). If checked, violations are Receiver Errors, and are reported errors associated with the Port (see Section 6.2).

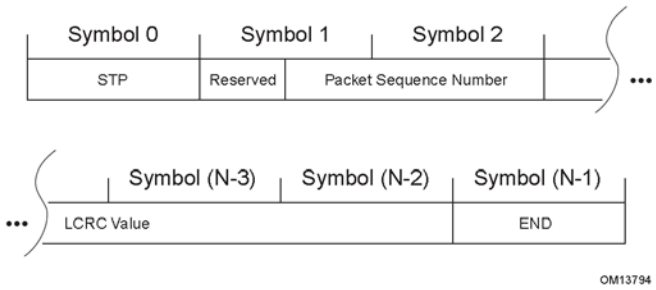


Figure 4-5: TLP with Framing Symbols Applied

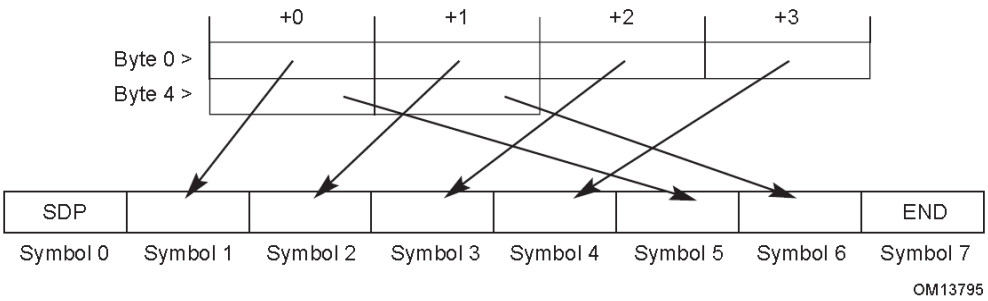


Figure 4-6: DLLP with Framing Symbols Applied

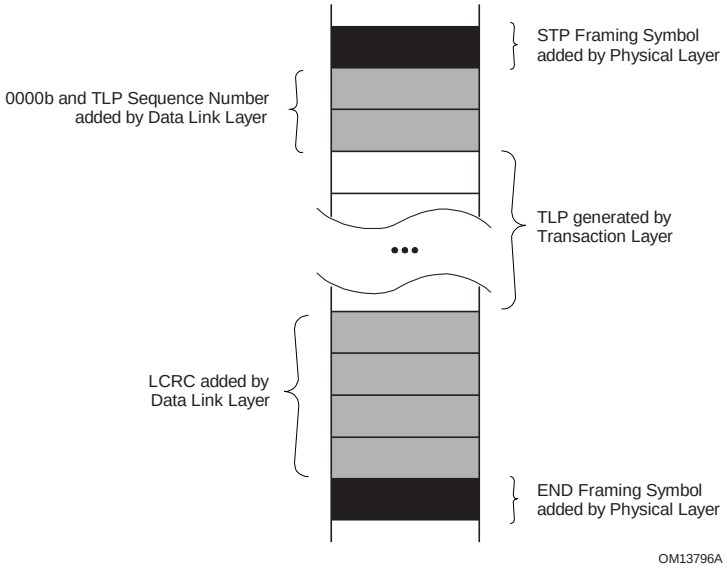


Figure 4-7: Framed TLP on a x1 Link

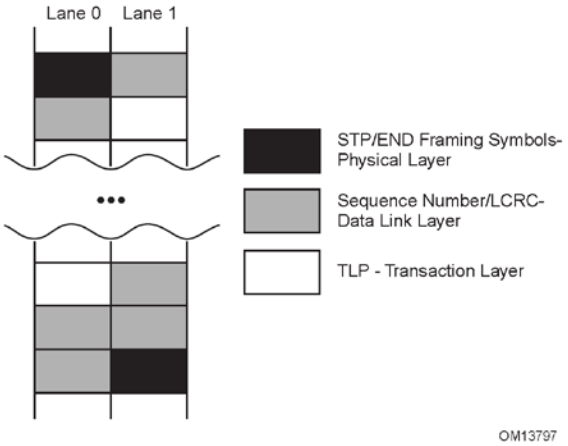


Figure 4-8: Framed TLP on a x2 Link

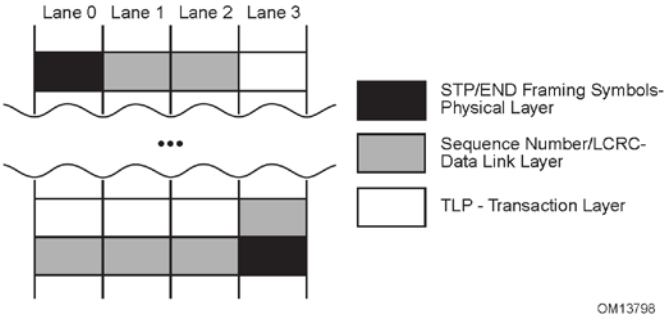


Figure 4-9: Framed TLP on a x4 Link

4.2.1.3. Data Scrambling

The scrambling function can be implemented with one or many Linear Feedback Shift Registers (LFSRs) on a multi-Lane Link. When there is more than one Transmit LFSR per Link, these must operate in concert, maintaining the same simultaneous (see Table 4-18, Lane-to-Lane Output Skew) value in each LFSR. When there is more than one Receive LFSR per Link, these must operate in concert, maintaining the same simultaneous (see Table 4-24, Lane-to-Lane Skew) value in each LFSR. Regardless of how they are implemented, LFSRs must interact with data on a Lane-by-Lane basis as if there was a separate LFSR as described here for each Lane within that Link. On the Transmit side, scrambling is applied to characters prior to the 8b/10b encoding. On the Receive side, de-scrambling is applied to characters after 8b/10b decoding.

The LFSR is graphically represented in Figure 4-10. Scrambling or unscrambling is performed by serially XORing the 8-bit (D0-D7) character with the 16-bit (D0-D15) output of the LFSR. An output of the LFSR, D15, is XORed with D0 of the data to be processed. The LFSR and data register are then serially advanced and the output processing is repeated for D1 through D7. The LFSR is advanced after the data is XORed. The LFSR implements the polynomial:

$G(X)=X^{16}+X^5+X^4+X^3+1$

The mechanism(s) and/or interface(s) utilized by the Data Link Layer to notify the Physical Layer to disable scrambling is implementation specific and beyond the scope of this specification.

The data scrambling rules are the following:

- The COM Symbol initializes the LFSR.
- The LFSR value is advanced eight serial shifts for each Symbol except the SKP.
- All data Symbols (D codes) except those within ~~Training Sequence~~ Ordered Sets (e.g., TS1, TS2, EIEOS), the Compliance Pattern (see Section 4.2.8), and the Modified Compliance Pattern (see Section 4.2.9) are scrambled.
- All special Symbols (K codes) are not scrambled.
- The initialized value of an LFSR seed (D0-D15) is FFFFh. Immediately after a COM exits the Transmit LFSR, the LFSR on the Transmit side is initialized. Every time a COM enters the Receive LFSR on any Lane of that Link, the LFSR on the Receive side is initialized.
- Scrambling can only be disabled at the end of Configuration (see Section 4.2.6.3.5).
- Scrambling does not apply to a loopback slave.
- Scrambling is always enabled in Detect by default.

Comment [VS103]: Errata B25



IMPLEMENTATION NOTE

Disabling Scrambling

Disabling scrambling is intended to help simplify test and debug equipment. Control of the exact data patterns is useful in a test and debug environment. Since scrambling is reset at the Physical Layer there is no reasonable way to reliably control the state of the data transitions through software. Thus, the Disable Scrambling bit in the TS1 and TS2 Ordered Sets is provided for these purposes.

The mechanism(s) and/or interface(s) utilized by the Data Link Layer to notify the Physical Layer to disable scrambling is implementation specific and beyond the scope of this specification.

For more information on scrambling, see Appendix C.

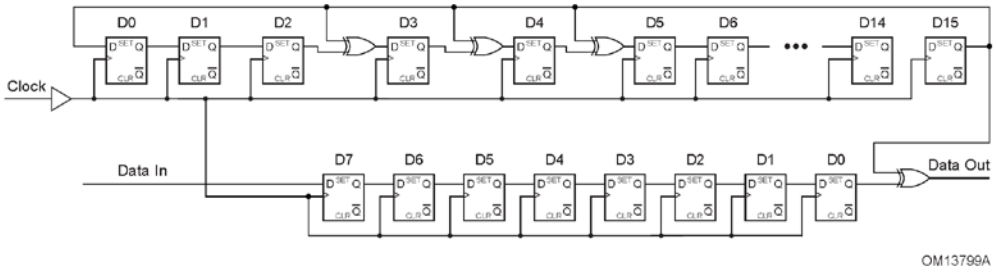


Figure 4-10: LFSR with Scrambling Polynomial

4.2.2. Encoding for 8.0 GT/s and Higher Data Rates

When a PCI Express Link is operating at a data rate of 8.0 GT/s or higher, it uses the encoding rules described in this subsection: 128b/130b encoding. For backwards compatibility, the Link initially trains to L0 at the 2.5 GT/s data rate using 8b/10b encoding as described in Section 4.2.1, then when the data rate is changed to 8.0 GT/s or higher, 128b/130b encoding is used. 128b/130b encoding is a Link-wide packetization mechanism and a per-Lane block code with scrambling. The basic entity of data transmission is an 8-bit data character, referred to as a Symbol, as shown in Figure 4-11 and Figure 4-12.

IMPLEMENTATION NOTE

Symbol in 128b/130b Encoding Scheme

In the 128b/130b encoding scheme, the Symbol is one byte long, similar to the 10-bit Symbol of 8b/10b encoding.

4.2.2.1. Lane Level Encoding

The physical layer uses a per-Lane block code. Each Block consists of a 2-bit Sync Header and a payload. There are two valid Sync Header encodings: 10b and 01b. The Sync Header defines the type of payload that the Block contains.

A Sync Header of 10b indicates a Data Block. Each Data Block has a 128 bit payload, resulting in a Block size of 130 bits. The payload is a Data Stream described in Section 4.2.2.3.

A Sync Header of 01b indicates an Ordered Set Block. Each Ordered Set Block has a 128 bit payload, resulting is a Block size of 130 bits except for the SKP Ordered Set which can be of variable length.

All Lanes of a multi-Lane Link must transmit Blocks with the same Sync Header simultaneously.

The bit transmission order is as follows. A Sync Header represented as `$H_1H_0$' is placed on a Lane starting with `H_0' and ending with `$H_1$'. A Symbol, represented as `$S_7S_6S_5S_4S_3S_2S_1S_0$', is placed on a Lane starting with `$S_0$' and ending with `S_7'. In the diagrams that show a time scale, bits represent the transmission order. In layout diagrams, bits are arranged in little-endian format, consistent with packet layout diagrams in other chapters of this specification.

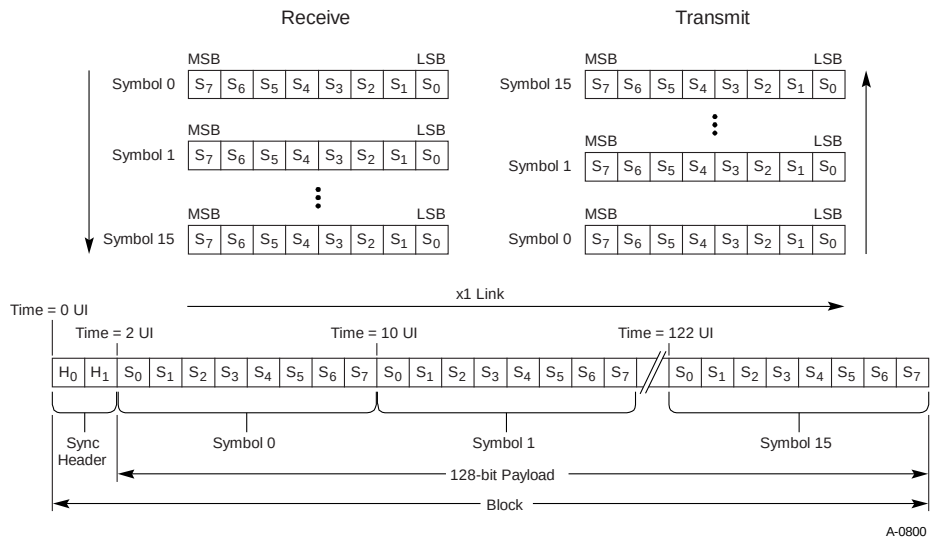


Figure 4-11: Example of Bit Transmission Order in a x1 Link Showing 130 Bits of a Block

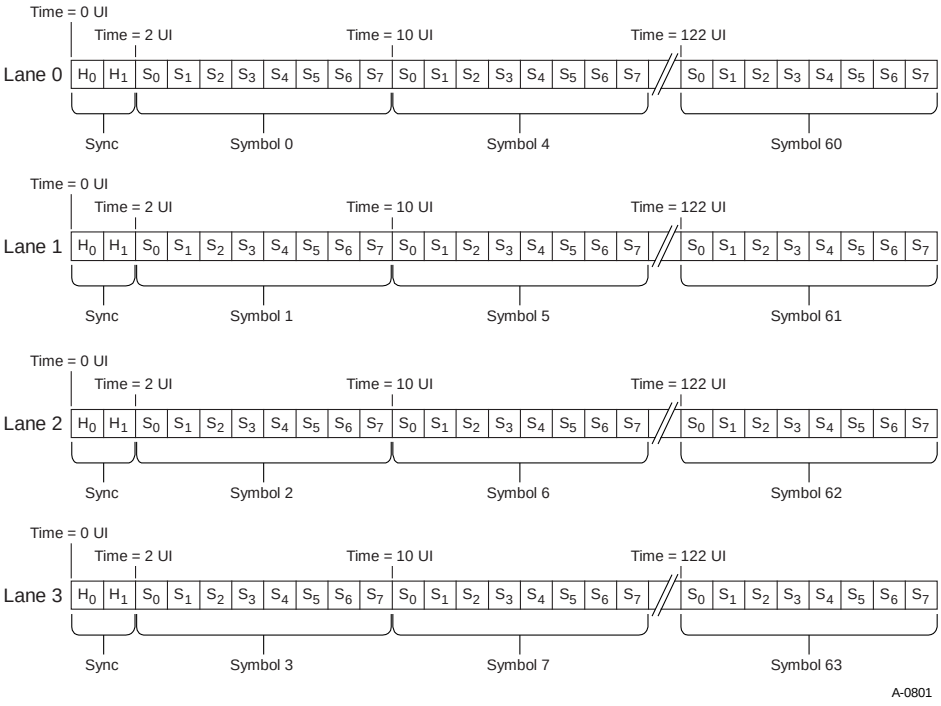


Figure 4-12: Example of Bit Placement in a x4 Link with One Block per Lane

4.2.2.2. Ordered Set Blocks

An Ordered Set Block contains a Sync Header followed by one Ordered Set. All Lanes of a multi-Lane Link must transmit the same Ordered Set simultaneously. The first Symbol of the Ordered Set defines the type of Ordered Set. The Ordered Sets are described in detail in Section 4.2.4 and Section 4.2.7.

4.2.2.2.1. Block Alignment

During Link training, the 130 bits of the Electrical Idle Exit Ordered Set (EIEOS) are a unique bit pattern that Receivers use to determine the location of the Block Sync Headers in the received bit stream. Conceptually, Receivers can be in three different phases of Block alignment: Unaligned, Aligned, and Locked. These phases are defined to illustrate the required behavior, but are not meant to specify a required implementation.

Unaligned Phase: Receivers enter this phase after a period of Electrical Idle, such as when the data rate is changed to one that uses 128b/130b encoding or when they exit a low-power Link state, or if directed. In this phase, Receivers monitor the received bit stream for the EIEOS bit pattern. When one is detected, they adjust their alignment to it and proceed to the Aligned phase.

Aligned Phase: Receivers monitor the received bit stream for the EIEOS bit pattern and the received Blocks for a Start of Data Stream (SDS) Ordered Set. If an EIEOS bit pattern is detected on an alignment that does not match the current alignment, Receivers must adjust their alignment to the newly received EIEOS bit pattern. If an SDS Ordered Set is received, Receivers proceed to the Locked phase. Receivers are permitted to return to the Unaligned phase if an undefined Sync Header (00b or 11b) is received.

Locked Phase: Receivers must not adjust their Block alignment while in this phase. Data Blocks are expected to be received after an SDS Ordered Set, and adjusting the Block alignment would interfere with the processing of these Blocks. Receivers must return to the Unaligned or Aligned phase if an undefined Sync Header is received.



IMPLEMENTATION NOTE

Detection of Loss of Block Alignment

The sequence of EIEOS and TS Ordered Sets transmitted during training sequences will cause misaligned Receivers to detect an undefined Sync Header.

Additional Requirements:

- While in the Aligned or Locked phase, Receivers must adjust their alignment as necessary when a SKP Ordered Set is received. See Section 4.2.7 for more information on SKP Ordered Sets.
- After any LTSSM transition to Recovery, Receivers must ignore all received TS Ordered Sets until they receive an EIEOS. Conceptually, receiving an EIEOS validates the Receiver's alignment and allows TS Ordered Set processing to proceed. If a received EIEOS initiates an LTSSM transition from L0 to Recovery, Receivers are permitted to process any TS Ordered Sets that follow the EIEOS or ignore them until another EIEOS is received after entering Recovery.

- Receivers are permitted to be directed from the Locked phase to the Unaligned or Aligned phase as long as Data Stream processing is stopped. See Section 4.2.2.3 for more information on Data Stream requirements.
- Loopback Masters: While in Loopback.Entry, Masters must be capable of adjusting their Receiver's Block alignment to received EIEOS bit patterns. While in Loopback.Active, Masters are permitted to transmit an EIEOS and adjust their Receiver's Block alignment to the looped back bit stream.
- Loopback Slaves: While in Loopback.Entry, Slaves must be capable of adjusting their Receiver's Block alignment to received EIEOS bit patterns. While in Loopback.Active, Slaves must not adjust their Receiver's Block alignment. Conceptually, the Receiver is directed to the Locked phase when the Slave starts to loop back the received bit stream.

4.2.2.3. Data Blocks

The payload of Data Blocks is a stream of Symbols defined as a "Data Stream" that consists of Framing Tokens, TLPs, and DLLPs. Each Symbol of the Data Stream is placed on a single Lane of the Link, and the stream of Symbols is striped across all Lanes of the Link and spans Block boundaries.

A Data Stream starts with the first Symbol of the Data Block that follows an SDS Ordered Set. It ends either when a Framing Error is detected or with the last Symbol of the Data Block that precedes an Ordered Set other than a SKP Ordered Set. SKP Ordered Sets that occur within a Data Stream have specific requirements as described in the following sections.

4.2.2.3.1. Framing Tokens

The Framing Tokens used by the Physical Layer are shown in Table 4-2. Each Framing Token specifies or implies the number of Symbols associated with the Token and therefore the location of the next Framing Token. Figure 4-15 shows an example of TLPs, DLLPs, and IDLs transmitted on a x8 link.

The first Framing Token of a Data Stream is always located in Symbol 0 of Lane 0 of the first Data Block of the Data Stream. For the rest of this chapter, the terms Framing Token and Token are used interchangeably.

Table 4-2: Framing Token Encoding

Framing Token Type	Description
IDL	Logical Idle. The Framing Token is 1 Symbol. This Token is transmitted when no TLPs or DLLPs or other Framing Tokens are being transmitted.
SDP	Start of DLLP. The Framing Token is 2 Symbols long and is followed by the ~~Data Link Layer~~ DLLP information.
STP	Start of TLP. The Framing Token is 4 Symbols long and includes the 12-bit TLP Sequence Number. It is followed by the ~~Data Link Layer~~ TLP information.
EDB	EnD Bad. The Framing Token is 4 Symbols long and is used to confirm that the previous TLP was nullified.
EDS	End of Data Stream. The Framing Token is four Symbols long and indicates that the next Block will be an Ordered Set Block.

Comment [VS104]: B9

Comment [VS105]: B9

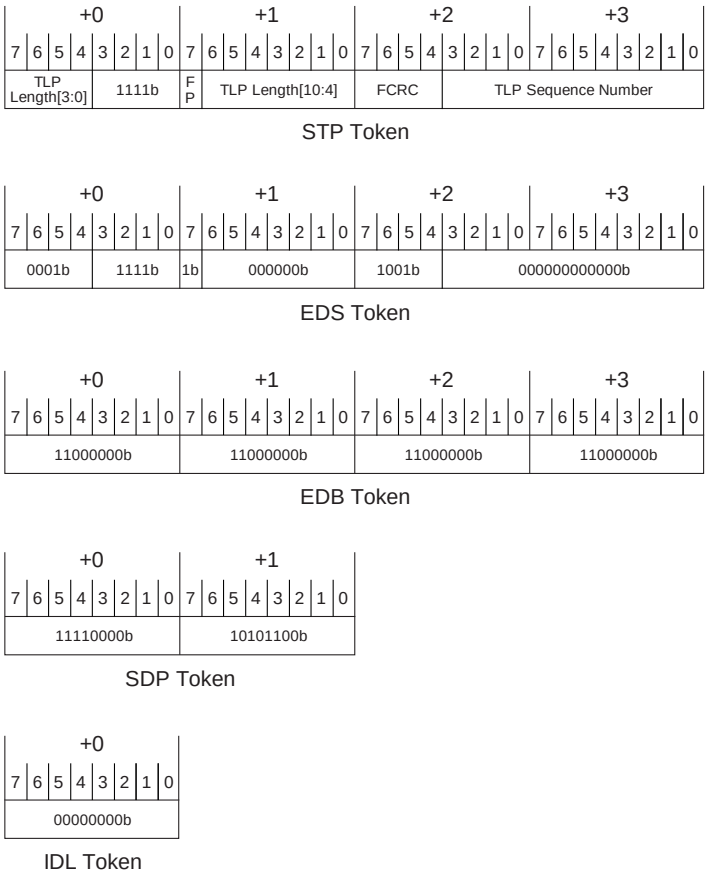


Figure 4-13: Layout of Framing Tokens

The Physical Layer DLLP layout is shown in Figure 4-14. Symbols 0 and 1 are the SDP Token, and Symbols 2 through 7 are the Data Link Layer DLLP information.

The Physical Layer TLP layout is shown in Figure 4-14. Details of the STP Framing Token are shown in Figure 4-13. The length of the TLP (in DWs) being transmitted is specified by an 11-bit field called TLP Length. The TLP Length field is the total amount of information transferred, including the Framing Token, TLP Prefixes (if any), TLP Header, TLP data payload (if any), TLP digest (if any), and TLP LCRC. For example, if a TLP has a 3 DW header, a 1 DW data payload, and does not include a TLP digest, the TLP Length field value is 6: 1 (Framing Token) + 0 (TLP Prefixes) + 3 (TLP header) + 1 (TLP data payload) + 0 (TLP digest) + 1 (TLP LCRC). If the same TLP included a TLP digest, the TLP Length field value would be 7. When a TLP is nullified, the EDB Token is considered an extension of the TLP but is not included in the calculation of the TLP Length field.

The TLP Length field is protected by a 4-bit CRC (Frame CRC), and an even parity bit (Frame Parity) protects both the TLP Length and Frame CRC fields. The Frame CRC and Frame Parity are calculated as follows:

C[0] = L[10] ^ L[7] ^ L[6] ^ L[4] ^ L[2] ^ L[1] ^ L[0]

C[1] = L[10] ^ L[9] ^ L[7] ^ L[5] ^ L[4] ^ L[3] ^ L[2]

C[2] = L[9] ^ L[8] ^ L[6] ^ L[4] ^ L[3] ^ L[2] ^ L[1]

C[3] = L[8] ^ L[7] ^ L[5] ^ L[3] ^ L[2] ^ L[1] ^ L[0]

P = L[10] ^ L[9] ^ L[8] ^ L[7] ^ L[6] ^ L[5] ^ L[4] ^ L[3] ^ L[2] ^ L[1] ^ L[0] ^ C[3] ^ C[2] ^ C[1] ^ C[0]

The Frame Parity reduces to P = L[10] ^ L[9] ^ L[8] ^ L[6] ^ L[5] ^ L[2] ^ L[0]

The TLP Length field is represented in the above equations as L[10:0], where L[0] is the least significant bit and L[10] is the most significant bit. Transmitters calculate the Frame CRC and Frame Parity before transmission. Receivers must calculate the Frame CRC and Frame Parity using the same algorithm as the transmitter and then compare the calculated values to the received values.

STP Tokens do not have a TLP Length field value of 1. If a received sequence of Symbols matches the format of an STP Token with a TLP Length field value of 1, the Symbols are evaluated to determine whether they match the EDS Token.



IMPLEMENTATION NOTE

Frame CRC and Frame Parity

The Frame CRC bits are effectively calculated as $(L[0] X^{14} + L[1] X^{13} + \ldots + L[9] X^{5} + L[10] X^{4})$ mod $(X^4 + X + 1)$. It should be noted that $X^4 + X + 1$ is a primitive polynomial and the CRC can detect two bit errors. The Frame Parity bit can detect an odd number of bit errors. Thus, the Frame CRC and Frame Parity together guarantee three bit error detection for the TLP Length field. It must be noted that even though in the reduced Frame Parity equation all terms are not present, it still maintains the property of detecting odd bit errors. Only those TLP Length field bits which are present in an even number of CRC terms are used in the calculation.

Note that, for TLPs, the Data Link Layer prepends 4 Reserved bits (0000b) to the TLP Sequence Number field before it calculates the LCRC. These Reserved bits are not explicitly transmitted when using 128b/130b encoding, and Receivers assume that the 4 bits received are 0000b when calculating the LCRC.

Packets containing a TLP Length field that is greater than 1535 are PMUX Packets. For such packets, the actual packet length is computed differently, the TLP Sequence Number field in the STP Token contains other information, and the Link CRC is computed using different rules. See Appendix G for details.

Packets containing a TLP Length field that is between 1152 and 1535 (inclusive) are reserved for future standardization.

Transmitters must transmit all DWs of a TLP specified by the TLP Length field of the STP Framing Token. TLPs are never truncated when using 128b/130b encoding - even when nullified. Figure 4-16 shows an example of a nullified 23 DW TLP.

Figure 4-17 shows an example of TLPs, DLLPs, IDLs, and an EDS Token followed by a SKP Ordered Set. SKP Ordered Sets are defined in Section 4.2.7.2.

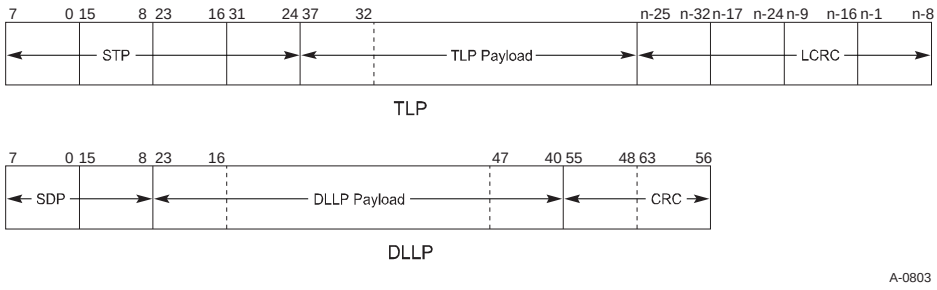


Figure 4-14: TLP and DLLP Layout

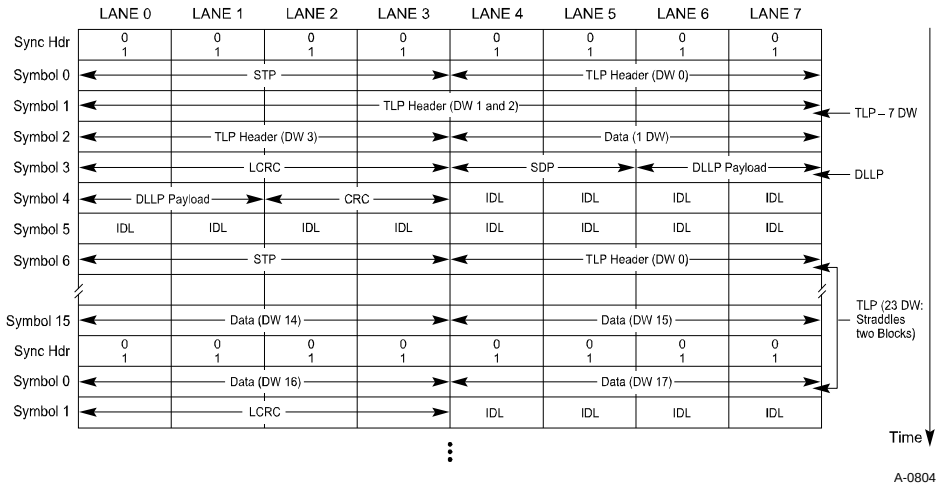


Figure 4-15: Packet Transmission in a x8 Link

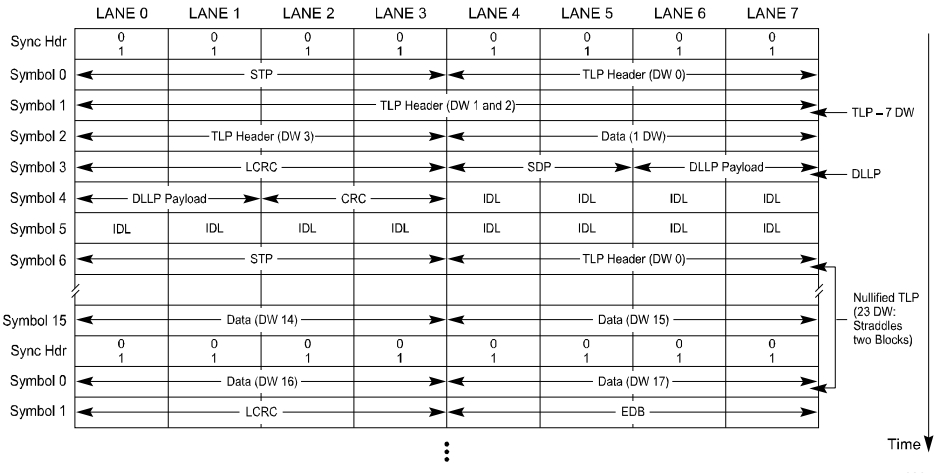


Figure 4-16: Nullified TLP Layout in a x8 Link with Other Packets

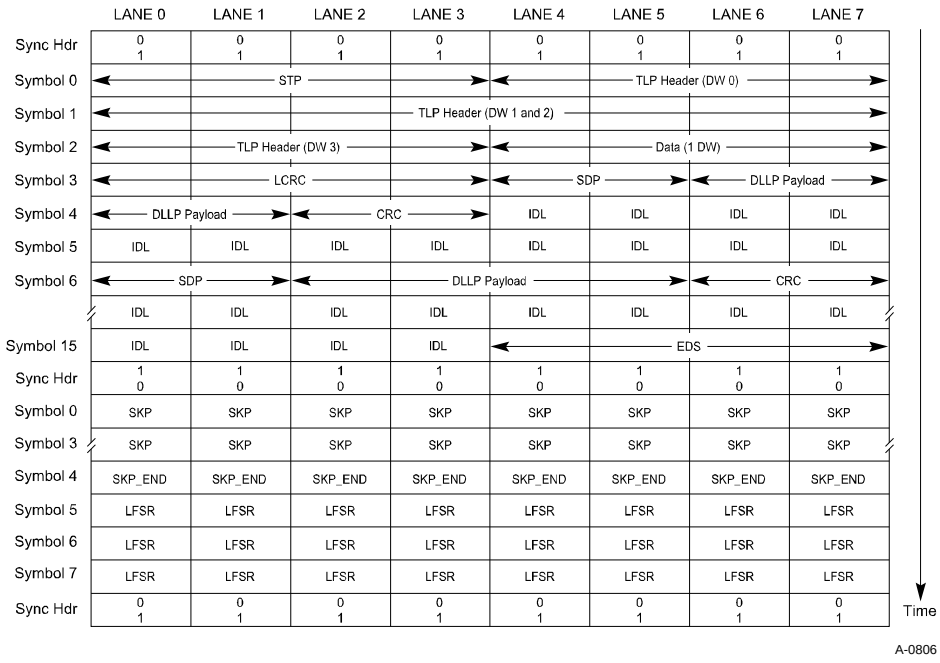


Figure 4-17: SKP Ordered Set of Length 66-bit in a x8 Link

4.2.2.3.2. Transmitter Framing Requirements

The following requirements apply to the transmitted Data Stream.

- To Transmit a TLP:
 - Transmit an STP Token immediately followed by the complete TLP information provided by the Data Link Layer.
 - All DWs of the TLP, as specified by the TLP Length field of the STP Token, must be transmitted, even if the TLP is nullified.
 - If the TLP is nullified, an EDB Token must be transmitted immediately following the TLP. There must be no Symbols between the last Symbol of the TLP and the first Symbol of the EDB Token. The value of the TLP Length field of a nullified TLP's STP Token is NOT adjusted to account for the EDB Token.
 - The STP Token must not be transmitted more frequently than once per Symbol Time.
- To Transmit a DLLP:
 - Transmit an SDP Token immediately followed by the complete DLLP information provided by the Data Link Layer.
 - All 6 Symbols of the DLLP must be transmitted.
 - The SDP Token must not be transmitted more frequently than once per Symbol Time.
- To Transmit a SKP Ordered Set within a Data Stream:
 - Transmit an EDS Token in the last DW of the current Data Block. For example, the Token is transmitted on Lane 0 in Symbol Times 12-15 of the Block for a x1 Link, and on Lanes 12-15 of Symbol Time 15 of the Block for a x16 Link.
 - Transmit the SKP Ordered Set following the current Data Block.
 - Transmit a Data Block following the SKP Ordered Set. The Data Stream resumes with the first Symbol of the Data Block. If multiple SKP Ordered Sets are scheduled for transmission, each SKP Ordered Set must be preceded by a Data Block with an EDS Token.
- To end a Data Stream:
 - Transmit an EDS Token in the last DW of the current Data Block, followed in the next block by an EIOS or an EIEOS. An EIOS is transmitted for LTSSM power management state transitions, and an EIEOS is transmitted for all other cases. For example, the Token is transmitted on Lane 0 in Symbol Times 12-15 of the Block for a x1 Link, and on Lanes 12-15 of Symbol Time 15 of the Block for a x16 Link.
- The IDL Token must be transmitted on all Lanes when not transmitting a TLP, DLLP, or other Framing Token.

- Multi-Lane Links:
 - After transmitting an IDL Token, the first Symbol of the next STP or SDP Token must be transmitted in Lane 0 of a future Symbol Time. An EDS Token can be transmitted after an IDL Token in the same Symbol Time, since it must be transmitted in the last DW of a Block.
 - For xN Links where N is 8 or more, if an EDB Token, TLP, or DLLP ends in a Lane K, where K does not equal N-1, and it is not followed by the first Symbol of an STP, SDP, or EDB Token in Lane K+1, then IDL Tokens must be placed in Lanes K+1 to N-1. For example, on a x8 Link, if a TLP or DLLP ends in Lane 3, IDL Tokens must be placed in Lanes 4 to 7. The EDS Token is an exception to this requirement, and can be transmitted following IDL Tokens.
 - Tokens, TLPs, and DLLPs are permitted to follow each other successively such that more than one Token may be transmitted in the same Symbol Time as long as their transmission conforms with the other requirements stated in this section.
 - Links wider than x4 can have Tokens placed starting on Lane 4*N, where N is a positive integer. For example, Tokens can be placed in Lanes 0 and 4 of a x8 Link, and Tokens can be placed in Lanes 0, 4, 8, or 12 of a x16 Link.

4.2.2.3.3. Receiver Framing Requirements

The following requirements apply to the received Data Stream and the Block type transitions that occur at the beginning and end of the Data Stream.

- When processing Symbols that are expected to be a Framing Token, receiving a Symbol or sequence of Symbols that does not match the definition of a Framing Token is a Framing Error.
- All optional error checks and error reports in this section are independently optional (see Section 6.2.3.4).
- When an STP Token is received:
 - Receivers must calculate the Frame CRC and Frame Parity of the received TLP Length field and compare the results to the received Frame CRC and Frame Parity fields. A Frame CRC or Frame Parity mismatch is a Framing Error.
 - An STP Token with Framing Error is not considered part of a TLP for the purpose of reporting to the Data Link Layer.
 - Receivers may optionally check whether the TLP Length field has a value of 0. If checked, receiving a TLP Length field of 0 is a Framing Error.
 - Receivers may optionally check whether the TLP Length field has a value between 1152 and 1535 (inclusive). If checked, receiving such a TLP Length field is a Framing Error.
 - Receivers on Ports that do not support Protocol Multiplexing may optionally check whether the TLP Length field has a value greater than 1535. If checked, receiving such a TLP Length field is a Framing Error.

- Receivers on Ports that support Protocol Multiplexing, shall process STP Tokens with a TLP Length field that is greater than 1535 as the start of a PMUX Packet as defined in Appendix G.
- The Symbol immediately following the last DW of the TLP is the next Token to be processed.
- Receivers must evaluate the Symbol immediately following the last DW of the TLP and determine whether it is the first Symbol of an EDB Token and therefore whether the TLP is nullified. See the EDB Token requirements.
- Receivers may optionally check whether more than one STP Token is received in a single Symbol Time. If checked, receiving more than one STP Token in a single Symbol Time is a Framing Error

❑ When an EDB Token is received:

- If an EDB Token is received immediately following a TLP (there are no Symbols between the last Symbol of the TLP and the first Symbol of the EDB Token), receivers must inform the Data Link Layer that an EDB Token has been received. Receivers are permitted to inform the Data Link Layer that an EDB Token has been received after processing the first Symbol of the EDB Token or after processing any or all of the remaining Symbols of the EDB Token. Regardless of when they inform the Data Link Layer of a received EDB Token, Receivers must check all Symbols of the EDB Token. Receiving a Symbol that does not match the definition of an EDB Token is a Framing Error.
- Receiving an EDB Token at any time other than immediately following a TLP is a Framing Error.
- The Symbol immediately following the EDB Token is the next Token to be processed.

❑ When an EDS Token is received in the last four Symbols of the Data Block across the Link:

- Receivers must stop processing the Data Stream.
- Receiving an Ordered Set other than SKP, EIOS, or EIEOS in the Block following the EDS Token is a Framing Error.
- If a SKP Ordered Set is received in the Block following the EDS Token, Receivers resume Data Stream processing with the first Symbol of the Data Block that follows the SKP Ordered Set unless a Framing Error has been detected.

❑ When an SDP Token is received:

- The Symbol immediately following the last Symbol of the DLLP is the next Token to be processed.
- Receivers may optionally check whether more than one SDP Token is received in a single Symbol Time. If checked, receiving more than one SDP Token in a single Symbol Time is a Framing Error.

❑ When an IDL Token is received:

- For a x1 Link, the next Token to be processed is the next Symbol received.

- For a x2 Link, the next Token to be processed is the Symbol received in Lane 0 of the next Symbol Time. Receivers may optionally check whether the Symbol received in Lane 1 after an IDL Token is also IDL. If checked, receiving a Symbol other than IDL is a Framing Error.
- For a x4 Link, the next Token to be processed is the Symbol received in Lane 0 of the next Symbol Time. Receivers may optionally check whether the Symbols received in Lanes 1-3 after an IDL Token are also IDL. If checked, receiving a Symbol other than IDL is a Framing Error.
- For x8, x12, x16, and x32 Links, the next Token to be processed is the Symbol received in the next DW aligned Lane following the IDL Token. For example, if an IDL Token is received in Lane 4 of a x16 Link, the next Token location is Lane 8 of the same Symbol Time. However, if an IDL Token is received on Lane 4 of a x8 Link, the next Token location is Lane 0 of the following Symbol Time. Receivers may optionally check whether the Symbols received between the IDL Token and the next Token location are also IDL. If checked, receiving a Symbol other than IDL is a Framing Error.

 Note: The only Tokens expected to be received in the same Symbol Time following an IDL Token are additional IDL Tokens or an EDS Token.

- While processing the Data Stream, Receivers must also check the Block type received by each Lane, after accounting for Lane-to-Lane de-skew, for the following conditions:
 - Receiving an Ordered Set Block on any Lane immediately following an SDS Ordered Set is a Framing Error.
 - Receiving a Block with an undefined Block type (a Sync Header of 11b or 00b) is a Framing Error. The Receiver may optionally report the error in the Lane Error Status register.
 - Receiving an Ordered Set Block on any Lane without receiving an EDS token in the preceding Block is a Framing Error. For example, receiving a SKP Ordered Set without a preceding EDS Token is a Framing Error. In addition, receiving a SKP Ordered Set followed immediately by another Ordered Set Block (including another SKP Ordered Set) within a Data Stream is a Framing Error.
 - Receiving a Data Block on any Lane when the previous block contained an EDS Token is a Framing Error.
 - Receivers may optionally check for different Ordered Sets on different Lanes. For example, Lane 0 receives a SKP Ordered Set and Lane 1 receives an EIOS. If checked, receiving different Ordered Sets is a Framing Error.

4.2.2.3.4. Recovery from Framing Errors

If a receiver detects a Framing Error while processing the Data Stream, it must:

- Report a Receiver Error as described in Section 4.2.4.7.
- Stop processing the Data Stream. Processing of a new Data Stream is initiated when the next SDS Ordered Set is received as previously described.

- Initiate the error recovery process as described in Section 4.2.4.7. If the LTSSM state is L0, direct the LTSSM to Recovery. If the LTSSM state is Configuration.Complete or Configuration.Idle when the Framing Error is detected, the error recovery process is satisfied by either a transition from Configuration.Idle to Recovery.RcvrLock due to the specified timeout, or a directed transition from L0 to Recovery. If the LTSSM state is Recovery.RcvrCfg or Recovery.Idle when the Framing Error is detected, the error recovery process is satisfied by either a transition from Recovery.Idle to Recovery.RcvrLock due to the specified timeout, or a directed transition from L0 to Recovery. If the LTSSM substate is either Recovery.RcvrLock or Configuration.Linkwidth.Start, the error recovery process is satisfied upon exit from these substates and no direction of the LTSSM to Recovery is required;
 - Note: The framing error recovery mechanism is not expected to directly cause any Data Link Layer initiated recovery action such as NAK.



IMPLEMENTATION NOTE

Time Spent in Recovery Due to Detection of a Framing Error

When using 128b/130b encoding, all Framing Errors require Link recovery. It is expected that implementations will require less than 1 microsecond to recover from a Framing Error as measured from the time that both Ports have entered the Recovery state.

4.2.2.4. Scrambling

Each Lane of the transmitter in a multi-Lane Link may implement a separate LFSR for scrambling. Each Lane of the receiver in a multi-Lane Link may implement a separate LFSR for descrambling. Implementations may choose to implement fewer LFSRs, but must achieve the same functionality as independent LFSRs.

The LFSR uses the following polynomial: $G(X) = X^{23} + X^{21} + X^{16} + X^{8} + X^{5} + X^{2} + 1$ and is demonstrated in Figure 4-18.

The scrambling rules are as follows:

- The two bits of the Sync Header are not scrambled and do not advance the LFSR.
- All 16 Symbols of an Electrical Idle Exit Ordered Set (EIEOS) bypass scrambling. The scrambling LFSR is initialized after the last Symbol of an EIEOS is transmitted, and the descrambling LFSR is initialized after the last Symbol of an EIEOS is received.
- TS1 and TS2 Ordered Sets:
 - Symbol 0 of a TS1 or TS2 Ordered Set bypasses scrambling.
 - Symbols 1-13 are scrambled.
 - Symbols 14 and 15 bypass scrambling if required for DC Balance, but they are scrambled if not required for DC Balance.
- All 16 Symbols of a Fast Training Sequence (FTS) Ordered Set bypass scrambling.
- All 16 Symbols of a Start of Data Stream (SDS) Ordered Set bypass scrambling.
- All 16 Symbols of an Electrical Idle Ordered Set (EIOS) bypass scrambling.

- All Symbols of a SKP Ordered Set bypass scrambling.
- Transmitters advance their LFSR for all Symbols of all Ordered Sets except for the SKP Ordered Set. The LFSR is not advanced for any Symbols of a SKP Ordered Set.
- Receivers evaluate Symbol 0 of Ordered Set Blocks to determine whether to advance their LFSR. If Symbol 0 of the Block is SKP (see Section 4.2.7.2), then the LFSR is not advanced for any Symbol of the Block. Otherwise, the LFSR is advanced for all Symbols of the Block.
- All 16 Symbols of a Data Block are scrambled and advance the scrambler.
- For Symbols that need to be scrambled, the least significant bit is scrambled first and the most significant bit is scrambled last.
- The seed value of the LFSR is dependent on the Lane number assigned to the Lane when the Link first entered Configuration.Idle (i.e., having gone through Polling from Detect with LinkUp = 0b).
 - The seed values for Lane number modulo 8 are:
 - 0: 1DBFBCh
 - 1: 0607BBh
 - 2: 1EC760h
 - 3: 18C0DBh
 - 4: 010F12h
 - 5: 19CFC9h
 - 6: 0277CEh
 - 7: 1BB807h



IMPLEMENTATION NOTE

Scrambling Pseudo-code

The pseudo-code for the scrambler along with examples is provided in Section C.2 of Appendix C.

- The seed value of the LFSR does not change while LinkUp=1. Link reconfiguration through the LTSSM Configuration state does not modify the initial Lane number assignment as long as the Link is Up (even though the Lane assignment may change during Configuration).
- Scrambling cannot be disabled in Configuration.Complete when using 128b/130b encoding.
- A Loopback Slave must not descramble or scramble the looped-back bit stream.

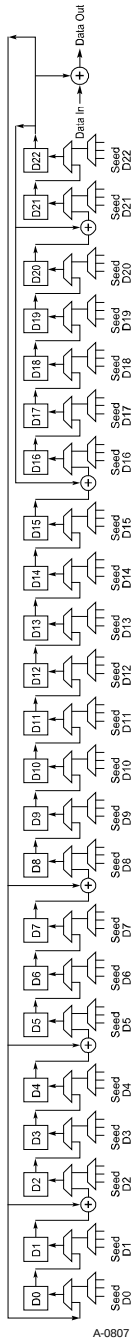


Figure 4-18: LFSR with Scrambling Polynomial in 8.0 GT/s and Above Data Rate



IMPLEMENTATION NOTE

LFSR Implementation with a Shared LFSR

Implementations may choose to implement one LFSR and take different tap points as shown in Figure 4-19, which is equivalent to the individual LFSR per-lane with different seeds, as shown in Figure 4-18. It should also be noted that the tap equations of four Lanes are the XOR of the tap equations of two neighboring Lanes. For example, Lane 0 can be obtained by XORing the output of Lanes 1 and 7; Lane 2 is the XOR of Lanes 1 and 3; Lane 4 is the XOR of Lanes 3 and 5; and Lane 6 is the XOR of Lanes 5 and 7. This can be used to help reduce the gate count at the expense of potential delay due to the XOR results of the two Lanes.

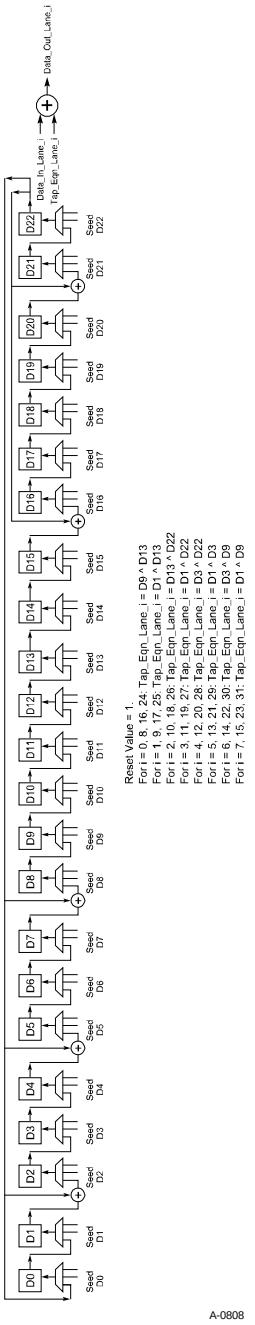


Figure 4-19: Alternate Implementation of the LFSR for Descrambling

4.2.2.5. Loopback with 128b/130b Code

When using 128b/130b encoding, Loopback Masters must transmit Blocks with the defined 01b and 10b Sync Headers. However, they are not required to transmit an SDS Ordered Set when transitioning from Ordered Set Blocks to Data Blocks, nor are they required to transmit an EDS Token when transitioning from Data Blocks to Ordered Set Blocks. Masters must transmit SKP Ordered Sets periodically as defined in Section 4.2.7, and they must be capable of processing received (looped-back) SKP Ordered Sets of varying length. Masters are permitted to transmit Electrical Idle Exit Ordered Sets (EIEOS) as defined in Section 4.2.2.2.1. Masters are permitted to transmit any payload in Data Blocks and Ordered Set Blocks that they expect to be looped-back. However, Ordered Set Blocks that match the definition of SKP OS, EIEOS, or EIOS should be avoided since they have defined purposes while in Loopback.

When using 128b/130b encoding, Loopback Slaves must retransmit all bits received without modification, except for SKP Ordered Sets which can be adjusted as needed for clock compensation. If clock compensation is required, slaves must add or remove 4 SKP Symbols per Ordered Set. The modified SKP Ordered Set must meet the definition of Section 4.2.7.2 (i.e., it must have between 4 to 20 SKP Symbols followed by the SKP_END Symbol and the three Symbols that follow it as transmitted by the Loopback Master). If a slave is unable to obtain Block alignment or it is misaligned, it may be unable to perform clock compensation and therefore unable to loop-back all bits received. In this case, it is permitted to add or remove Symbols as necessary to continue operation. Slaves must not check for a received SDS Ordered Set when a transition from Ordered Set Blocks to Data Blocks is detected, and they must not check for a received EDS Token when a transition from Data Blocks to Ordered Set Blocks is detected.

4.2.3. Link Equalization Procedure for 8.0 GT/s Data Rate

The Link equalization procedure enables components to adjust the Transmitter and the Receiver setup of each Lane to improve the signal quality and meet the requirements specified in Section 4.3, when operating at 8.0 GT/s data rate. All the Lanes that are associated with the LTSSM (i.e., those Lanes that are currently operational or may be operational in the future due to Link Upconfigure) must participate in the Equalization procedure. The procedure must be executed during the first data rate change to 8.0 GT/s. Components must arrive at the appropriate Transmitter setup for all the operating conditions they will encounter in the future when LinkUp=1b. Components must not require that the equalization procedure be repeated for reliable operation, although there is provision to repeat the procedure. Components must store their Transmitter setup that was agreed to during the equalization procedure and use it for future operation at 8.0 GT/s data rate. Components are permitted to fine-tune their Receiver setup even after the equalization procedure is complete as long as doing so does not cause the Link to be unreliable (i.e., does not meet the requirements in Section 4.3) or go to Recovery.

The equalization procedure can be initiated either autonomously or by software. It is strongly recommended that components use the autonomous mechanism. However, a component that chooses not to participate in the autonomous mechanism must have its associated software ensure that the software based mechanism is applied.

The autonomous mechanism is executed if both components advertise that they are capable of 8.0 GT/s data rate (via the TS1 and TS2 Ordered Sets) during the initial Link negotiation (when LinkUp is set to 1b). After entering L0, irrespective of the current Link speed, neither component

must transmit any DLLP if the equalization procedure must be performed and until the equalization procedure completes. The Downstream Port is required to make the transition from L0 to Recovery to change the data rate to 8.0 GT/s and perform the equalization procedure; the Upstream Port is permitted, but not required, to make the transition from L0 to Recovery autonomously. Components using the autonomous mechanism must not initiate any autonomous Link width downsizing until the equalization procedure completes. An Upstream Port must not transmit any DLLP until it receives a DLLP from the Downstream Port. If the Downstream Port performs equalization again, it must not transmit any DLLP until it completes the equalization procedure. A Downstream Port may perform equalization again based on its own needs or based on the request from the Upstream Port, if it can meet its system requirements. Executing equalization multiple times may interfere with software determination of Link and device status, as described in Section 6.6.



IMPLEMENTATION NOTE

DLLP Blocking During Autonomous Equalization

When using the autonomous mechanism for equalization at 8.0 GT/s, the Downstream Port is required to block the transmission of DLLPs until equalization has completed, and the Upstream Port is required to block the transmission of DLLPs until a DLLP is received from the Downstream Port. If the Downstream Port delays entering Recovery from L0 while DLLP transmission is blocked, either the L0 Inferred Electrical Idle timeout (see Section 4.2.4.3) or the DLLP timeout (see Section 2.6.1.2) may expire in the Upstream or Downstream Ports. If either of these two timeouts occurs, it will result in the initiation of an entry to Recovery to perform Link retraining. Neither of these two timeouts is a reportable error condition, and the resulting Link retraining has no impact on proper Link operation.

When using the software based mechanism, software must guarantee that there will be no side-effects for transactions in flight (e.g., no timeout), if any, due to the Link undergoing the equalization procedure. Software can write 1b to the Perform Equalization bit in the Link Control 3 register, followed by a write to the Target Link Speed field in the Link Control 2 register to enable the Link to run at 8.0 GT/s, followed by a write of 1b to the Retrain Link bit in the Link Control register of the Downstream Port to perform equalization. Software may set the Hardware Autonomous Width Disable of the Link Control register in both components or use some other mechanism to ensure that the Link is in its full functional width prior to setting the Perform Equalization bit in the Downstream Port. The component that had initiated the autonomous width downsizing is responsible to upconfigure the Link to go to its full functional width by initiating the transition to Recovery and Configuration within 1 ms of the Hardware Autonomous Width Disable bit being set to 1b. If an Upstream Port does not advertise 8.0 GT/s data rate initially and participate in the autonomous equalization mechanism, its associated software must ensure there will be no side-effects for transactions in flight, if any, during equalization, before it instructs the Upstream Port to go to Recovery and advertise the 8.0 GT/s data rate and initiate a speed change. The Downstream Port subsequently initiates the equalization procedure during the initial speed change to 8.0 GT/s when it transitions to Recovery.

Upstream Ports are required to check for equalization setting problems in the Recovery.RcvrLock state (see Section 4.2.6.4.1). However, both Downstream and Upstream Ports are permitted to use

implementation-specific methods to detect equalization problems at any time. A Port that detects problems with its equalization setting must request equalization be redone and set the Link Equalization Request register bit of its Link Status 2 register to 1b even if it may decide not to operate at 8.0 GT/s data rate. An Upstream Port must set this bit, transition to Recovery, and request equalization using the Request Equalization bit in the TS2 Ordered Sets it sends in the Recovery.RcvrCfg substate. It should request equalization only once for each problem detected. When an Upstream Port requests equalization be redone through its TS2 Ordered Sets, a Downstream Port is required to either redo the equalization if it can ensure that there will be no side-effects for transactions in flight, if any, or set its Link Equalization Request register bit to 1b. The Downstream Port may redo equalization by re-entering Recovery within 1 ms of entering L0 from Recovery, if the Request Equalization and Quiesce Guarantee bits are set to 1b in the eight consecutive TS2 Ordered Sets it received in Recovery. An Upstream Port is required to ensure that there will be no side-effects due to redoing equalization if it sets the Request Equalization and Quiesce Guarantee bits to 1b in the TS2 Ordered Sets. If the Downstream Port wants to redo equalization, it may also request the Upstream Port to make a re-equalization request. In order for the Downstream Port to make this request, it sets the Request Equalization and Quiesce Guarantee bits to 1b in the TS2 Ordered Sets in Recovery.RcvrCfg. The Upstream Port may eventually respond with a re-entry to Recovery with the Request Equalization and Quiesce Guarantee bits set to 1b if it is capable of doing so.



IMPLEMENTATION NOTE

Using Quiesce Guarantee Mechanism

Side-effects due to executing equalization after the Data Link Layer is in DL_Active can occur at the Port, Device, or system level. For example, the time required to execute the equalization process could cause a Completion Timeout error to occur – possibly in a different system component. The Quiesce Guarantee information can help Ports decide whether to execute a requested equalization or not.

A component may operate at a lower data rate after reporting its equalization problems, either by timing out through Recovery.Speed or by initiating a data rate change to a lower data rate. Any data rate change required to perform equalization procedure is exempt from the 200 ms requirement in Section 6.11. If the Downstream Port wants to redo equalization and the current data rate is either 2.5 GT/s or 5.0 GT/s, the Downstream Port must request speed change through Equalization (EQ) TS1 Ordered Sets in Recovery.RcvrLock to inform the Upstream Port that it intends to redo equalization. An Upstream Port should advertise 8.0 GT/s data rate in Recovery if it receives EQ TS1 Ordered Sets with speed change bit set to 1b and if it intends to operate at 8.0 GT/s data rate in the future. The equalization procedure can be performed at most once in each trip through the Recovery state.

The equalization procedure consists of up to four Phases, as described below. When operating at the 8.0 GT/s data rate, the Phase information is transmitted using the Equalization Control (EC) field in the TS1 Ordered Sets.

Phase 0: The Downstream Port communicates the Transmitter preset values and the Receiver preset hints for each Lane to the Upstream Port using 8b/10b encoding. These values are communicated using the EQ TS2 Ordered Sets (defined in Section 4.2.4.1) in Recovery.RcvrCfg, when a data rate change to 8.0 GT/s has been negotiated, prior to transitioning to 8.0 GT/s data

rate. These preset values are derived from the Upstream Port Transmitter Preset and Upstream Port Receiver Preset Hint fields of each Lane's Equalization Control register. After the data rate change to 8.0 GT/s, the Upstream Port transmits TS1 Ordered Sets with the preset values it received. The preset values must be within the operable range defined in Section 4.3.3.5.2 if reduced swing will be used by the Transmitter.

Phase 1: Both components make the Link operational enough at 8.0 GT/s data rate to be able to exchange TS1 Ordered Sets to complete remaining phases for the fine-tuning their Transmitter/Receiver pairs. It is expected that the Link will operate at a BER of less than 10^{-4} before the component is ready to move on to the next Phase.

The Downstream Port initiates Phase 1 by transmitting TS1 Ordered Sets with EC=01b (indicating Phase 1) to the Upstream Port using the preset values in the Downstream Port Transmitter Preset and, optionally, the Downstream Port Receiver Preset Hint fields of each Lane's Equalization Control register. The Upstream Port, after adjusting its Receiver, if necessary, to ensure that it can progress with the equalization process, receives these TS1 Ordered Sets and transitions to Phase 1 (where it transmits TS1 Ordered Sets with EC=01b). The Downstream Port ensures that it can reliably receive the bit stream from the Upstream Port to continue through the rest of the Phases when it receives TS1 Ordered Sets from the Upstream Port with EC=01b before it moves on to Phase 2.

Phase 2: In this Phase the Upstream Port adjusts the Transmitter setting of the Downstream Port along with its own Receiver setting, independently, on each Lane, to ensure it receives the bit stream compliant with the requirements in Section 4.3 (e.g., all operational Downstream Lanes have a BER is less than 10^{-12}). The Downstream Port initiates the move to Phase 2 by transmitting TS1 Ordered Sets with EC=10b to the Upstream Port. The Downstream Port advertises the Transmitter coefficients and the preset it is using per the rules below in Phase 1 for preset only and in Phase 2 for preset and coefficients. The Upstream Port receives these Ordered Sets and may request different coefficient or preset settings and continue to evaluate each setting until it arrives at the best setting for operating the Downstream Lanes. After the Upstream Port has completed this Phase, it moves the Link to Phase 3 by transmitting TS1 Ordered Sets with EC=11b to the Downstream Port.

Phase 3: In this Phase the Downstream Port adjusts the Transmitter setting of the Upstream Port along with its own Receiver setting, independently, on each Lane, using a handshake and evaluation process similar to Phase 2 with the exception that EC=11b. The Downstream Port signals the end of Phase 3 (and the equalization procedure) by transmitting TS1 Ordered Sets with EC=00b.

The algorithm used by a component to adjust the transmitter of its Link partner and the evaluation of that Transmitter set-up with its Receiver set-up is implementation-specific. A component may request changes to any number of Lanes and can request different settings for each Lane. Each requested setting can be a preset or a set of coefficients that meets the requirements defined in Section 4.2.3.1. Each component is responsible for ensuring that at the end of the fine-tuning (Phase 2 for Upstream Ports and Phase 3 for Downstream Ports), its Link partner has the Transmitter setting in each Lane that will cause the Link to meet the requirements in Section 4.3.

A Link partner receiving the request to adjust its Transmitter must evaluate the request and act on it. If a valid preset value is requested and the Transmitter is operating in full-swing mode, it must be reflected in the Transmitter set-up and subsequently in the preset and coefficient fields of the TS1 Ordered Set that the Link partner transmits. If a preset value is requested, the Transmitter is

operating in reduced-swing mode, and the requested preset is supported as defined in Section 4.3.3.5.2 it must be reflected in the Transmitter set-up and subsequently in the preset and coefficient fields of the TS1 Ordered Set that the Link partner transmits. Transmitters operating in reduced-swing mode are permitted to reject preset requests that are not supported as defined in Section 4.3.3.5.2. A request for adjusting the coefficients may be accepted or rejected. If the set of coefficients requested for a Lane is accepted, it must be reflected in the Transmitter set-up and subsequently in the transmitted TS1 Ordered Sets. If the set of coefficients requested for a Lane is rejected, the Transmitter set-up is not changed, but the transmitted TS1 Ordered Sets must reflect the requested coefficients along with the Reject Coefficient bit set to 1b. In either case of responding to a coefficient request, the preset field of the transmitted TS1 Ordered Sets is not changed from the last preset value that was transmitted. A request for adjusting the coefficients may be rejected by the Link partner only if the set of coefficients requested is not compliant with the rules defined in Section 4.2.3.1.

When performing equalization of a crosslink, the component that played the role of the Downstream Port during the earlier crosslink initialization at the lower data rate also assumes the responsibility of the Downstream Port for equalization.

If a Lane is directed to use a Reserved or unsupported Transmitter Preset in Polling.Compliance, Loopback, or Phase 0 or Phase 1 of Recovery.Equalization, then the Lane is permitted to use any supported Transmitter Preset in an implementation-specific manner. The Reserved or unsupported Transmitter Preset value is transmitted in any subsequent Compliance Patterns or Ordered Sets, and not the implementation-specific preset value chosen by the Lane. For example, if a Lane of an Upstream Port is directed to use Transmitter Preset 1111b (Reserved) with the EQ TS2 Ordered Sets it receives in Recovery.RcvrCfg, it is permitted to use any supported Transmitter Preset for its transmitter setting after changing the data rate to 8.0 GT/s, but it must transmit 1111b as its Transmitter Preset value in the TS1 Ordered Sets it transmits in Phase 0 and Phase 1 of Recovery.Equalization.

In the Loopback state, the Loopback Master is responsible for communicating the Transmitter and Receiver settings it wants the Slave to use through the EQ TS1 Ordered Sets it transmits in the 2.5 GT/s or 5.0 GT/s data rate, and the preset or coefficient settings it wants the device under test to operate under in the TS1 Ordered Sets it transmits in the 8.0 GT/s data rate. Similarly, if the Polling.Compliance state is entered through TS1 Ordered Sets, the entity that is performing the test is required to send the appropriate EQ TS1 Ordered Sets and coefficients for the device under test to operate with, according to the mechanism defined in Section 4.2.6.2.

IMPLEMENTATION NOTE

Equalization Example

The following diagram is an example illustrating how two devices may complete the equalization procedure. If equalization procedure is repeated and is performed while the Link is in 8.0 GT/s data rate, Phase 0 may be skipped since there is no need for the Link to go back to 2.5 GT/s or 5.0 GT/s to resend the same EQ TS2 Ordered Sets to convey the presets. A Downstream Port may choose to skip Phase 2 and Phase 3 if it determines that fine-tuning of the Transmitter is not needed based on the channel and components in the platform.

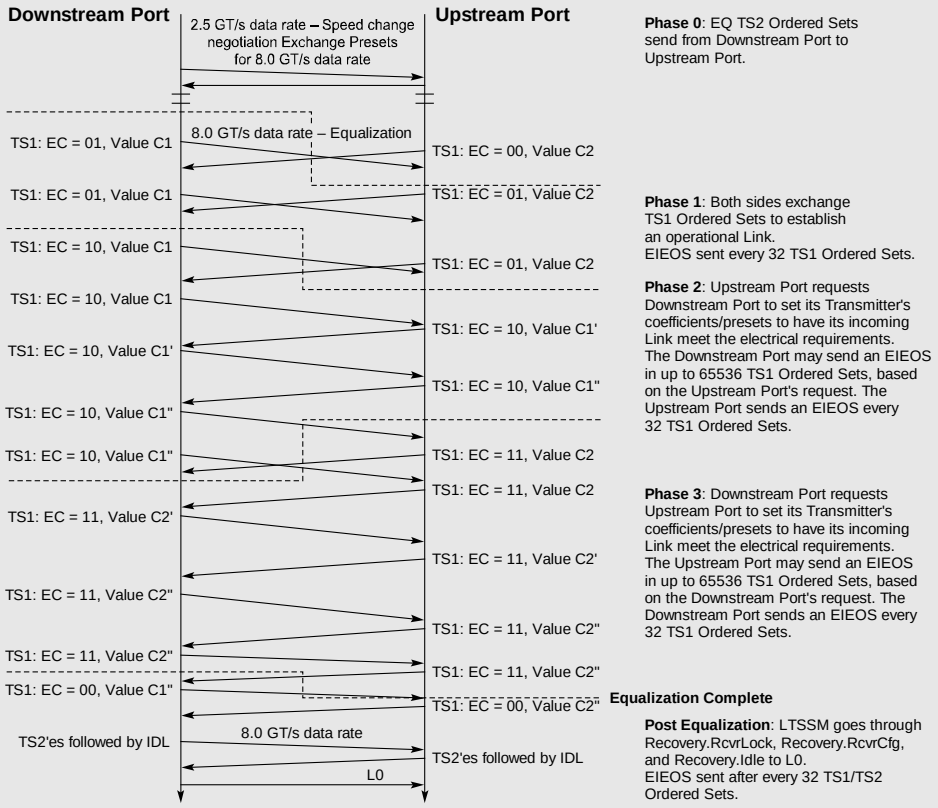


Figure 4-20: Equalization Flow

4.2.3.1. Rules for Transmitter Coefficients

The explanation of the coefficients and the FIR filter it represents are provided in Section 4.3.3.5.1. The following rules apply to both the advertised as well as requested coefficient settings.

1. C_{-1} and C_{+1} are the coefficients used in the FIR equation and represent the pre-cursor and post-cursor, respectively. The pre-cursor and post-cursor values communicated in the TS1 Ordered Sets represent their absolute values. C_0 represents the cursor coefficient setting and is a positive entity.
2. The sum of the absolute values of the coefficients defines the FS (Full Swing; FS = $|C_{-1}|+C_0+|C_{+1}|$). FS is advertised to the Link partner in Phase 1. The Transmitter FS range is defined below:
 - FS ~~∈~~ {24, …, 63} (i.e., FS must have a value from 24 through 63) for full swing mode.
 - FS ~~∈~~ {12, …, 63} for reduced swing mode.
3. A Transmitter advertises its LF (Low Frequency) value during Phase 1. This corresponds to the minimum differential voltage that can be generated by the Transmitter which is LF/FS times the Transmitters maximum differential voltage. The Transmitter must ensure that when equation c) below equals LF it must meet the electrical requirements defined in Section 4.3.3.5.5 for $V_{TX-EIEOS-FS}$ and $V_{TX-EIEOS-RS}$.
4. The following rules must be satisfied before a set of coefficients can be requested of the Link partner's Transmitter. Upon reception of an update request for TX coefficient settings, a Port must verify that the new request meets the following conditions and reject the request if any of following conditions are violated:
 a) $|C_{-1}| <=$ Floor (**FS**/4)
 b) $|C_{-1}|+C_0+|C_{+1}| =$ **FS** (Do not allow peak power to change with adaptation.)
 c) $C_0-|C_{-1}|-|C_{+1}| >=$ **LF**

Comment [VS106]: Errata B12

Comment [VS107]: Errata B12

4.2.3.2. Encoding of Presets

Definition of the Transmitter and Receiver presets appears in Section 4.3. The encoding for the Transmitter and Receiver presets are provided in Table 4-3 and Table 4-4. ~~The Transmitter Preset encoding of 1010b corresponds to the maximum Vtx-boost that the Transmitter advertises with FS and LF in both full swing and reduced swing modes with $C_{-1} = 0$ and $C_{+1} = (FS - LF)/2$.~~

Comment [VS108]: Errata B14

Comment [VS109]: B14

Table 4-3: Transmitter Preset Encoding

Encoding	Preset Number in Table 4-16
0000b	P0
0001b	P1
0010b	P2
0011b	P3
0100b	P4
0101b	P5
0110b	P6
0111b	P7
1000b	P8
1001b	P9
1010b	P10
1011b through 1111b	Reserved

Table 4-4: Receiver Preset Hint Encoding

Encoding	Receiver Preset Value
000b	-6 dB
001b	-7 dB
010b	-8 dB
011b	-9 dB
100b	-10 dB
101b	-11 dB
110b	-12 dB
111b	Reserved

4.2.4. Link Initialization and Training

This section defines the Physical Layer control process that configures and initializes each Link for normal operation. This section covers the following features:

- configuring and initializing the Link

- supporting normal packet transfers
- supported state transitions when recovering from Link errors
- restarting a Port from low power states.

The following are discovered and determined during the training process:

- Link width
- Link data rate
- Lane reversal
- Lane polarity

Training does:

- Link data rate negotiation.
- Bit lock per Lane
- Lane polarity
- Symbol lock or Block alignment per Lane
- Lane ordering within a Link
- Link width negotiation
- Lane-to-Lane de-skew within a multi-Lane Link.

4.2.4.1. Training Sequences

Training sequences are composed of Ordered Sets used for initializing bit alignment, Symbol alignment and to exchange Physical Layer parameters. When the data rate is 2.5 GT/s or 5.0 GT/s, ~~training sequence~~ Ordered Sets are never scrambled but are always 8b/10b encoded. When the data rate is 8.0 GT/s or higher, the 128b/130b encoding is used and Symbols may or may not be scrambled, according to the rules in Section 4.2.2.4.

Comment [VS110]: Errata B25

Training sequences (TS1 or TS2) are transmitted consecutively and can only be interrupted by SKP Ordered Sets (see Section 4.2.7) or EIEOSs in data rates other than 2.5 GT/s (see below). When 8.0 GT/s data rate is supported, a TS1 (or TS2) Ordered Set using 8b/10b encoding (i.e., 2.5 or 5.0 GT/s data rate) can be either a standard TS1 (or TS2) Ordered Set (i.e., Symbol 6 is D10.2 for a TS1 Ordered Set or D5.2 for a TS2 Ordered Set) or an EQ TS1 (or EQ TS2) (i.e., Symbol 6 bit 7 is 1b). The ability to transmit EQ TS1 Ordered Sets is implementation-specific. Ports supporting 8.0 GT/s data rate must accept either TS1 (or TS2) type in the LTSSM states unless explicitly required to look for a specific type. Ports that do not support the 8.0 GT/s data rate are permitted, but not required, to accept EQ TS1 (or TS2) Ordered Sets.

When using 8b/10b encoding, TS1 or TS2 Ordered Sets are considered consecutive only if Symbol 6 matches the Symbol 6 of the previous TS1 or TS2 Ordered Set.

When using 128b/130b encoding, TS1 or TS2 Ordered Sets are considered consecutive only if Symbols 6-9 match Symbols 6-9 of the previous TS1 or TS2 Ordered Set, with Reserved bits treated as described below.

Reserved bits in TS1 and TS2 Ordered Sets must be handled as follows:

- The Transmitter must transmit 0s for Reserved bits.
- The Receiver:
 - must not determine that a TS1 or TS2 Ordered Set is invalid based on the received value of Reserved bits
 - must use the received value of Reserved bits for the purpose of a parity computation if the Reserved bits are included in a parity calculation
 - may optionally compare the received value of Reserved bits within Symbols that are explicitly called out as being required to be identical in TS1 or TS2 Ordered Sets to determine if they are consecutive
 - must not otherwise take any functional action based on the value of any received Reserved bits

When using 128b/130b encoding, Transmitters are required to track the running DC Balance of the bits transmitted on the wire (after scrambling) for TS1 and TS2 Ordered Sets. The running DC Balance is the difference between the number of 1s transmitted and the number of 0s transmitted. Each Lane must track its running DC Balance independently and be capable of tracking a difference of at least 511 bits in either direction: 511 more 1s than 0s, and 511 more 0s than 1s. Any counters used must saturate at their limit (not roll-over) and continue to track reductions after their limit is reached. For example, a counter that can track a difference of 511 bits will saturate at 511 if a difference of 513 is detected, and then change to 509 if the difference is reduced by 2 in the future.

The running DC Balance is set to 0 by two events: 1) The Transmitter exiting Electrical Idle; 2) Transmission of an EIEOS following a Data Block.

For every TS1 or TS2 Ordered Set transmitted, Transmitters must evaluate the running DC Balance and transmit one of the DC Balance Symbols defined for Symbols 14 and 15 as defined by the algorithm below. If the number of 1s needs to be reduced, the DC Balance Symbols 20h (for Symbol 14) and 08h (for Symbol 15) are transmitted. If the number of 0s needs to be reduced, the DC Balance Symbols DFh (for Symbol 14) and F7h (for Symbol 15) are transmitted. If no change is required, the appropriate TS1 or TS2 Identifier Symbol is transmitted. Any DC Balance Symbols transmitted for Symbols 14 or 15 bypass scrambling, while TS1 and TS2 Identifier Symbols follow the standard scrambling rules. The following algorithm must be used to control the DC Balance:

- If the running DC Balance is > 31 at the end of Symbol 11 of the TS Ordered Set, transmit DFh for Symbol 14 and F7h for Symbol 15 to reduce the number of 0s, or 20h for Symbol 14 and 08h for Symbol 15 to reduce the number of 1s.
- Else, if the running DC Balance is > 15 at the end of Symbol 11 of the TS Ordered Set, transmit F7h for Symbol 15 to reduce the number of 0s, or 08h for Symbol 15 to reduce the number of 1s. Transmit the normal TS Identifier Symbol (scrambled) for Symbol 14.
- Else, transmit the normal TS Identifier Symbol (scrambled) for Symbols 14 and 15.

Receivers are permitted, but not required, to check Symbols 14 and 15 for the following values when determining whether a TS1 or TS2 Ordered Set is valid: The appropriate TS Identifier Symbol after de-scrambling, or a valid DC Balance Symbol of DFh or 20h before de-scrambling for Symbol 14, or a valid DC Balance Symbol of F7h or 08h before de-scrambling for Symbol 15.

IMPLEMENTATION NOTE

Sync Header and DC Balance

Block Sync Header bits and the first Symbol of TS1 and TS2 Ordered Sets do not affect the running DC Balance, because they have equal number of 1s and 0s.

The Training control bits for Hot Reset, Disable Link, and Enable Loopback are mutually exclusive, only one of these bits is permitted to be set at a time as well as transmitted on all Lanes in a configured (all Lanes that were in L0) or possible (all Lanes in Configuration) Link. If more than one of the Hot Reset, Disable Link, or Enable Loopback bits are set at the same time, the Link behavior is undefined.

Table 4-5: TS1 Ordered Set

Symbol Number	Description
0	When operating at 2.5 or 5.0 GT/s: COM (K28.5) for Symbol alignment. When operating at 8.0 GT/s or above: Encoded as 1Eh (TS1 Ordered Set).
1	Link Number. Ports that do not support 8.0 GT/s or above: 0-255, PAD. Downstream Ports that support 8.0 GT/s or above: 0-31, PAD. Upstream Ports that support 8.0 GT/s or above: 0-255, PAD. When operating at 2.5 or 5.0 GT/s: PAD is encoded as K23.7. When operating at 8.0 GT/s or above: PAD is encoded as F7h.
2	Lane Number within Link. When operating at 2.5 or 5.0 GT/s: 0-31, PAD. PAD is encoded as K23.7. When operating at 8.0 GT/s or above: 0-31, PAD. PAD is encoded as F7h.
3	N_FTS. The number of Fast Training Sequences required by the Receiver: 0-255.
4	Data Rate Identifier Bit 0 – Reserved Bit 1 – 2.5 GT/s Data Rate Supported. Must be set to 1b. Bit 2 – 5.0 GT/s Data Rate Supported. Must be set to 1b if Bit 3 is 1b. See Section 4.3.2. Bit 3 – 8.0 GT/s Data Rate Supported. Bit 4:5 – Reserved. Bit 6 – Autonomous Change/Selectable De-emphasis. Downstream Ports: This bit is defined for use in the following LTSSM states: Polling.Active, Configuration.LinkWidth.Start, and Loopback.Entry. In all other LTSSM states, it is Reserved. Upstream Ports: This bit is defined for use in the following LTSSM states: Polling.Active, Configuration, Recovery, and Loopback.Entry. In all other LTSSM states, it is Reserved. Bit 7 – speed_change. This bit can be set to 1b only in the Recovery.RcvrLock LTSSM state. In all other LTSSM states, it is Reserved.

Comment [VS111]: Errata A1

Symbol Number	Description
5	Training Control Bit 0 – Hot Reset Bit 0 = 0b, De-assert Bit 0 = 1b, Assert Bit 1 – Disable Link Bit 1 = 0b, De-assert Bit 1 = 1b, Assert Bit 2 – Loopback Bit 2 = 0b, De-assert Bit 2 = 1b, Assert Bit 3 – Disable Scrambling in 2.5 GT/s and 5.0 GT/s data rates; Reserved in other data rates Bit 3 = 0b, De-assert Bit 3 = 1b, Assert Bit 4 – Compliance Receive Bit 4 = 0b, De-assert Bit 4 = 1b, Assert Ports that support 5.0 GT/s and above data rate(s) must implement the Compliance Receive bit. Ports that support only 2.5 GT/s data rate may optionally implement the Compliance Receive bit. If not implemented, the bit is Reserved. Bit 5:7 – Reserved
6	When operating at 2.5 or 5.0 GT/s: Standard TS1 Ordered Sets encode this Symbol as a TS1 Identifier, D10.2 (4Ah). EQ TS1 Ordered Sets encode this Symbol as follows: Bit 2:0 – Receiver Preset Hint. See Section 4.2.3.2. Bit 6:3 – Transmitter Preset. See Section 4.2.3.2. Bit 7 – Set to 1b. When operating at 8.0 GT/s: Bit 1:0 – Equalization Control (EC). This field is only used in the Recovery.Equalization and Loopback LTSSM states . See Section 4.2.6.4.2 and Section 4.2.6.10. In all other LTSSM states, it must be set to 00b. Bit 2 – Reset EIEOS Interval Count. This bit is defined for use in the Recovery.Equalization LTSSM state. See Section 4.2.6.4.2 and Section 4.2.4.2. In all other LTSSM states, it is Reserved. Bit 6:3 – Transmitter Preset. See Section 4.2.3 and Section 4.2.6. Bit 7 – Use Preset. This bit is defined for use in the Recovery.Equalization and Loopback LTSSM states. See Section 4.2.6.4.2 and Section 4.2.6.10. In all other LTSSM states, it is Reserved.
7	When operating at 2.5 or 5.0 GT/s: TS1 Identifier. Encoded as D10.2 (4Ah). When operating at 8.0 GT/s: Bit 5:0 – FS when the EC field of Symbol 6 is 01b (see Section 4.2.3.1). Otherwise, Pre-cursor Coefficient. Bit 7:6 – Reserved.

Symbol Number	Description
8	When operating at 2.5 or 5.0 GT/s: TS1 Identifier. Encoded as D10.2 (4Ah). When operating at 8.0 GT/s: Bit 5:0 – LF when the EC field of Symbol 6 is 01b (see Section 4.2.3.1). Otherwise, Cursor Coefficient. Bit 7:6 – Reserved.
9	When operating at 2.5 or 5.0 GT/s: TS1 Identifier. Encoded as D10.2 (4Ah). When operating at 8.0 GT/s: Bit 5:0 – Post-cursor Coefficient. Bit 6 – Reject Coefficient Values. This bit can only be set to 1b in specific Phases of the Recovery.Equalization LTSSM State. See Section 4.2.6.4.2. In all other LTSSM states, it must be set to 0b. Bit 7 – Parity (P). This bit is the even parity of all bits of Symbols 6, 7, and 8 and bits 6:0 of Symbol 9. Receivers must calculate the parity of the received bits and compare it to the received Parity bit. Received TS1 Ordered Sets are valid only if the calculated and received Parity match.
10 – 13	When operating at 2.5 or 5.0 GT/s: TS1 Identifier. Encoded as D10.2 (4Ah). When operating at 8.0 GT/s or above: TS1 Identifier. Encoded as 4Ah.
14 – 15	When operating at 2.5 or 5.0 GT/s: TS1 Identifier. Encoded as D10.2 (4Ah). When operating at 8.0 GT/s or above: TS1 Identifier (encoded as 4Ah) or a DC Balance Symbol.

Table 4-6: TS2 Ordered Set

Symbol Number	Description
0	When operating at 2.5 or 5.0 GT/s: COM (K28.5) for Symbol alignment. When operating at 8.0 GT/s or above: Encoded as 2Dh (TS2 Ordered Set).
1	Link Number. Ports that do not support 8.0 GT/s or above: 0-255, PAD. Downstream Ports that support 8.0 GT/s or above: 0-31, PAD. Upstream Ports that support 8.0 GT/s or above: 0-255, PAD. When operating at 2.5 or 5.0 GT/s: PAD is encoded as K23.7. When operating at 8.0 GT/s or above: PAD is encoded as F7h.
2	Lane Number within Link. When operating at 2.5 or 5.0 GT/s: 0-31, PAD. PAD is encoded as K23.7. When operating at 8.0 GT/s or above: 0-31, PAD. PAD is encoded as F7h.
3	N_FTS. The number of Fast Training Sequences required by the Receiver: 0-255.
4	Data Rate Identifier Bit 0 – Reserved Bit 1 – 2.5 GT/s Data Rate Supported. Must be set to 1b. Bit 2 – 5.0 GT/s Data Rate Supported. Must be set to 1b if Bit 3 is 1b. See Section 4.3.2. Bit 3 – 8.0 GT/s Data Rate Supported. Bit 4:5 – Reserved. Bit 6 – Autonomous Change/Selectable De-emphasis/Link Upconfigure Capability. This bit is defined for use in the following LTSSM states: Polling.Configuration, Configuration.Complete, and Recovery. In all other LTSSM states, it is Reserved. Bit 7 – speed_change. This bit can be set to 1b only in the Recovery.RcvrCfg LTSSM state. In all other LTSSM states, it is Reserved.
5	Training Control Bit 0 – Hot Reset Bit 0 = 0b, De-assert Bit 0 = 1b, Assert Bit 1 – Disable Link Bit 1 = 0b, De-assert Bit 1 = 1b, Assert Bit 2 – Loopback Bit 2 = 0b, De-assert Bit 2 = 1b, Assert Bit 3 – Disable Scrambling in 2.5 GT/s and 5.0 GT/s data rates; Reserved in other data rates Bit 3 = 0b, De-assert Bit 3 = 1b, Assert Bit 4:7 – Reserved

Comment [VS112]: Errata A1

Symbol Number	Description
6	When operating at 2.5 or 5.0 GT/s: Standard TS2 Ordered Sets encode this Symbol as a TS2 Identifier, D5.2 (45h). EQ TS2 Ordered Sets encode this Symbol as follows: Bit 2:0 ~~–~~ Receiver Preset Hint. See Section 4.2.3.2. Bit 6:3 ~~–~~ Transmitter Preset. See Section 4.2.3.2. Bit 7 – ~~Equalization Command. See Section 4.2.6.4.4~~Set to 1b. When operating at 8.0 GT/s: Bit 5:0 – Reserved. Bit 6 – Quiesce Guarantee. This bit is defined for use in the Recovery.RcvrCfg LTSSM state. In all other LTSSM states, it is Reserved. Bit 7 – Request Equalization. This bit is defined for use in the Recovery.RcvrCfg LTSSM state. In all other LTSSM states, it is Reserved.
7 – 13	When operating at 2.5 or 5.0 GT/s: TS2 Identifier. Encoded as D5.2 (45h). When operating at 8.0 GT/s or above: TS2 Identifier. Encoded as 45h.
14-15	When operating at 2.5 or 5.0 GT/s: TS2 Identifier. Encoded as D5.2 (45h). When operating at 8.0 GT/s or above: TS2 Identifier (encoded as 45h) or a DC Balance Symbol.

Comment [VS113]: Errata B13

4.2.4.2. Electrical Idle Sequences

Before a Transmitter enters Electrical Idle, it must always send the Electrical Idle Ordered Set (EIOS), unless otherwise specified.

When using 8b/10b encoding, an EIOS is a K28.5 (COM) followed by three K28.3 (IDL) Symbols. Transmitters must transmit all Symbols of an EIOS. An EIOS is received when the COM and two of the three IDL Symbols are received. When using 128b/130b encoding, an EIOS is an Ordered Set block, as defined in Table 4-8. Transmitters must transmit all Symbols of an EIOS if additional EIOSs are to be transmitted following it. Transmitters must transmit Symbols 0-13 of an EIOS, but are permitted to terminate the EIOS anywhere in Symbols 14 or 15, when transitioning to Electrical Idle after it. An EIOS is considered received when Symbols 0-3 of an Ordered Set Block match the definition of an EIOS.



IMPLEMENTATION NOTE

Truncation of EIOS Ordered Set

Truncation in the last EIOS is allowed to help implementations where a transmitter may terminate on an internal clock boundary that may not align on a Symbol boundary due to 128b/130b encoding. Truncation is okay since Receivers will just look at the first four Symbols to conclude it is an EIOS.

After transmitting the last Symbol of the last Electrical Idle Ordered Set, the Transmitter must be in a valid Electrical Idle state as specified by $T_{TX\text{-}IDLE\text{-}SET\text{-}TO\text{-}IDLE}$ (see Table 4-18).

Table 4-7: Electrical Idle Ordered Set (EIOS) for 2.5 GT/s and 5.0 GT/s Data Rates

Symbol Number	Encoded Values	Description
0	K28.5	COM for Symbol alignment
1	K28.3	IDL
2	K28.3	IDL
3	K28.3	IDL

Table 4-8: Electrical Idle Ordered Set (EIOS) for 8.0 GT/s and Above Data Rates

Symbol Numbers	Value	Description
0-15	66h	EIOS Identifier and Payload

Table 4-9: Electrical Idle Exit Ordered Set (EIEOS) for 5.0 GT/s Data Rate

Symbol Number	Encoded Values	Description
0	K28.5	COM for Symbol alignment
1-14	K28.7	EIE – K Symbol with low frequency components for helping achieve exit from Electrical Idle
15	D10.2	TS1 Identifier (See Note 1)

Comment [VS114]: Errata B25

Notes:

1. This symbol is not scrambled. Previous versions of this specification were less clear and some implementations may have incorrectly scrambled this symbol. It is recommended that devices be tolerant of receiving EIEOS in which this symbol is scrambled.

Table 4-10: Electrical Idle Exit Ordered Set (EIEOS) for 8.0 GT/s and Above Data Rates

Symbol Numbers	Value	Description
0, 2, 4, 6, 8, 10, 12, 14	00h	Symbol 0: EIEOS Identifier A low frequency pattern that alternates between eight 0s and eight 1s.
1, 3, 5, 7, 9, 11, 13, 15	FFh	A low frequency pattern that alternates between eight 0s and eight 1s.

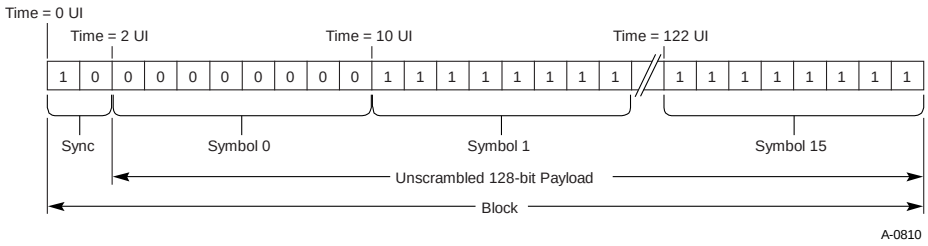


Figure 4-21: Electrical Idle Exit Ordered Set for 8.0 GT/s and Above Data Rates

Comment [VS115]: Errata B25

The Electrical Idle Exit Ordered Set (EIEOS) is transmitted only when operating at speeds other than 2.5 GT/s. It is a low frequency pattern transmitted periodically to help ensure that receiver Electrical Idle exit circuitry can detect an exit from Electrical Idle. When using 128b/130b encoding, it is also used for Block Alignment as described in Section 4.2.2.2.1.

When using 8b/10b encoding and operating at 5.0 GT/s, an EIEOS, as defined in Table 4-9, is transmitted ~~as follows~~ in the following situations:

Comment [VS116]: Errata B25

- ❑ Before the first TS1 Ordered Set after entering the LTSSM Configuration.Linkwidth.Start state.
- ❑ Before the first TS1 Ordered Set after entering the LTSSM Recovery.RcvrLock state.
- ❑ After every 32 TS1 or TS2 Ordered Sets are transmitted in the LTSSM Configuration.Linkwidth.Start, Recovery.RcvrLock, and Recovery.RcvrCfg states. The TS1/TS2 count is set to 0 when:
 - o An EIEOS is transmitted.
 - o The first TS2 Ordered Set is received while in the LTSSM Recovery.RcvrCfg state.

When using 128b/130b encoding, an EIEOS, as defined in Table 4-10 and Figure 4-21, is transmitted ~~as follows~~ in the following situations:

Comment [VS117]: Errata B25

- ❑ Before the first TS1 Ordered Set after entering the LTSSM Configuration.Linkwidth.Start substate.
- ❑ Before the first TS1 Ordered Set after entering the LTSSM Recovery.RcvrLock substate.
- ❑ Immediately following an EDS Framing Token when ending a Data Stream and not transmitting an EIOS and not entering the LTSSM Recovery.RcvrLock substate.
- ❑ After every 32 TS1 or TS2 Ordered Sets are transmitted in all LTSSM states which require transmission of TS1 or TS2 Ordered Sets. The TS1/TS2 count is set to 0 when:
 - o An EIEOS is transmitted.
 - o The first TS2 Ordered Set is received while in the LTSSM Recovery.RcvrCfg state.
 - o The first TS2 Ordered Set is received while in the LTSSM Configuration.Complete state.
 - o A Downstream Port is in Phase 2 of the LTSSM Recovery.Equalization state and two consecutive TS1 Ordered Sets are received on any Lane with the Reset EIEOS Interval Count bit set.

- An Upstream Port is in Phase 3 of the LTSSM Recovery.Equalization state and two consecutive TS1 Ordered Sets are received on any Lane with the Reset EIEOS Interval Count bit set.

- After every 65,536 TS1 Ordered Sets are transmitted in the LTSSM Recovery.Equalization state if the Reset EIEOS Interval Count bit has prevented it from being transmitted for that interval. Implementations are permitted to satisfy this requirement by transmitting an EIEOS within two TS1 Ordered Sets of when the scrambling LFSR matches its seed value.
- As part of an FTS Ordered Set, Compliance Pattern, or Modified Compliance pattern as described in the relevant sections.

Example: An LTSSM enters Recovery.RcvrLock from L0 in 5.0 GT/s data rate. It transmits an EIEOS followed by TS1 Ordered Sets. It transmits 32 TS1 Ordered Sets following which it transmits the second EIEOS. Subsequently it sends two more TS1 Ordered Sets and enters Recovery.RcvrCfg where it transmits the third EIEOS after transmitting 30 TS2 Ordered Sets. It transmits 31 more TS2 Ordered Sets (after the first 30 TS2 Ordered Sets) in Recovery.RcvrCfg when it receives a TS2 Ordered Set. Since it receives its first TS2 Ordered Set, it will reset its EIEOS interval count to 0 and keep transmitting another 16 TS2 Ordered Sets before transitioning to Recovery.Idle. Thus, it did not send an EIEOS in the midst of the last 47 TS2 Ordered Sets since the EIEOS interval count got reset to 0b. From Recovery.Idle, the LTSSM transitions to Configuration.Linkwidth.Start and transmits an EIEOS after which it starts transmitting the TS1 Ordered Sets.

While operating in speeds other than 2.5 GT/s, an implementation is permitted to not rely on the output of the Electrical Idle detection circuitry except when receiving the EIEOS during certain LTSSM states or during the receipt of the FTS prepended by the four consecutive EIE Symbols (see Section 4.2.4.5) at the Receiver during Rx L0s or the Modified Compliance Pattern in Polling.Compliance when the circuitry is required to signal an exit from Electrical Idle.

4.2.4.3. Inferring Electrical Idle

A device is permitted in all speeds of operation to infer Electrical Idle instead of detecting Electrical Idle using analog circuitry. Table 4-11 summarizes the conditions to infer Electrical Idle in the various substates.

Table 4-11: Electrical Idle Inference Conditions

State	2.5 GT/s	5.0 GT/s	8.0 GT/s
L0	Absence of Flow Control Update DLLP[42] or alternatively a SKP Ordered Set in a 128 µs window	Absence of Flow Control Update DLLP[42] or alternatively a SKP Ordered Set in a 128 µs window	Absence of Flow Control Update DLLP[42] or alternatively a SKP Ordered Set in a 128 µs window
Recovery.RcvrCfg	Absence of a TS1 or TS2 Ordered Set in a 1280 UI interval	Absence of a TS1 or TS2 Ordered Set in a 1280 UI interval	Absence of a TS1 or TS2 Ordered Set in a 4 ms window
Recovery.Speed when successful_speed_negotiation = 1b	Absence of a TS1 or TS2 Ordered Set in a 1280 UI interval	Absence of a TS1 or TS2 Ordered Set in a 1280 UI interval	Absence of a TS1 or TS2 Ordered Set in a 4680 UI interval
Recovery.Speed when successful_speed_negotiation = 0b	Absence of an exit from Electrical Idle in an 2000 UI interval	Absence of an exit from Electrical Idle in a 16000 UI interval	Absence of an exit from Electrical Idle in a 16000 UI interval
Loopback.Active (as slave)	Absence of an exit from Electrical Idle in a 128 µs window	N/A	N/A

The Electrical Idle exit condition must not be determined based on inference of Electrical Idle condition. For area efficiency, an implementation is permitted to choose to implement a common timeout counter per LTSSM and look for the Electrical Idle inference condition within the common timeout window determined by the common counter for each of the Lanes the LTSSM controls instead of having a timeout counter per Lane.

[42] A Flow Control Update DLLP is either an UpdateFC as defined in this specification or an MRUpdateFC as defined in the *Multi-Root I/O Virtualization and Sharing Specification* (MR-IOV).

IMPLEMENTATION NOTE

Inference of Electrical Idle

In the L0 state, some number of Flow Control Update DLLPs are expected to be received in a 128 µs window. Also in L0, a SKP Ordered Set is expected to be received on average every 1538 Symbols in 2.5 GT/s and 5.0 GT/s data rates and every 375 Blocks in 8.0 GT/s data rate. As a simplification, it is permitted to use either one (or both) of these indicators to infer Electrical Idle. Hence, the absence of either a Flow Control Update DLLP or alternatively a SKP Ordered Set in any 128 µs window can be inferred as Electrical Idle. In Recovery.RcvrCfg as well as Recovery.Speed with successful speed negotiation, the Receiver should receive TS1 or TS2 Ordered Sets continuously with the exception of the EIEOS and the SKP Ordered Set. Hence, the absence of a TS1 or TS2 Ordered Set in the interval specified above must be treated as Electrical Idle for components that implement the inference mechanism. In the event that the device enters Recovery.Speed with successful_speed_negotiation = 0b, there is a possibility that the device had failed to receive Symbols. Hence, the Electrical Idle inference is done as an absence of exit from Electrical Idle. In data rates other than 2.5 GT/s, Electrical Idle exit is guaranteed only on receipt of an EIEOS. Hence, the window is set to 16000 UI for detecting an exit from Electrical Idle in 5.0 GT/s and 8.0 GT/s data rates. In 2.5 GT/s data rate, Electrical Idle exit must be detected with every Symbol received. Hence, absence of Electrical Idle exit in a 2000 UI window constitutes an Electrical Idle condition.

4.2.4.4. Lane Polarity Inversion

During the training sequence in Polling, the Receiver looks at Symbols 6-15 of the TS1 and TS2 Ordered Sets as the indicator of Lane polarity inversion (D+ and D- are swapped). If Lane polarity inversion occurs, the TS1 Symbols 6-15 received will be D21.5 as opposed to the expected D10.2. Similarly, if Lane polarity inversion occurs, Symbols 6-15 of the TS2 Ordered Set will be D26.5 as opposed to the expected D5.2. This provides the clear indication of Lane polarity inversion.

If polarity inversion is detected the Receiver must invert the received data. The Transmitter must never invert the transmitted data. Support for Lane Polarity Inversion is required on all PCI Express Receivers across all Lanes independently.

4.2.4.5. Fast Training Sequence (FTS)

Fast Training Sequence (FTS) is the mechanism that is used for bit and Symbol lock when transitioning from L0s to L0. The FTS is used by the Receiver to detect the exit from Electrical Idle and align the Receiver's bit and Symbol receive circuitry to the incoming data. Refer to Section 4.2.5 for a description of L0 and L0s.

- At 2.5 GT/s and 5.0 GT/s data rates:
 A single FTS is comprised of one K28.5 (COM) Symbol followed by three K28.1 Symbols. The maximum number of FTSs (N_FTS) that a component can request is 255, providing a bit time lock of 4 * 255 * 10 * UI. If the data rate is 5.0 GT/s, four consecutive EIE Symbols are transmitted at valid signal levels prior to transmitting the first FTS. These Symbols will help the Receiver detect exit from Electrical Idle. An implementation that does not guarantee proper signaling levels for up to the allowable time on the Transmitter pins (see Table 4-18) since exiting Electrical Idle condition is required to prepend its first FTS by extra EIE Symbols so that the Receiver can receive at least four EIE Symbols at valid signal levels. Implementations must not transmit more than eight EIE Symbols prior to transmitting the first FTS. A component is permitted to advertise different N_FTS rates at different speeds. At 5.0 GT/s, a component may choose to advertise an appropriate N_FTS number considering that it will receive the four EIE Symbols. 4096 FTSs must be sent when the Extended Synch bit is set in order to provide external Link monitoring tools with enough time to achieve bit and framing synchronization. SKP Ordered Sets must be scheduled and transmitted between FTSs as necessary to meet the definitions in Section 4.2.7 with the exception that no SKP Ordered Sets can be transmitted during the first N_FTS FTSs. A single SKP Ordered Set is always sent after the last FTS is transmitted. It is permitted for this SKP Ordered Set to affect or not affect the scheduling of subsequent SKP Ordered Sets for Clock Tolerance Compensation by the Transmitter as described in Section 4.2.7. Note that it is possible that two SKP Ordered Sets can be transmitted back to back (one SKP Ordered Set to signify the completion of the 4096 FTSs and one scheduled and transmitted to meet the definitions described in Section 4.2.7).

 Comment [VS118]: Errata A5

- At 8.0 GT/s data rate:
 A single FTS is a 130-bit unscrambled Ordered Set Block, as shown in Table 4-12. The maximum number of FTSs (N_FTS) that a component can request is 255, providing a bit time lock of 130 * 255 UI (130 * 263 UI if including the periodic EIEOS). A component is permitted to advertise different N_FTS values at different speeds. On exit from L0s, the transmitter first transmits an EIEOS block which will help the receiver detect exit from Electrical Idle due to its low frequency content. After that first EIEOS, the transmitter must send the required number of FTS (4096 when the Extended Synch bit is set; otherwise N_FTS), with an EIEOS transmitted after every 32 FTS. The FTS sequence will enable the receiver obtain bit lock (and optionally to do Block alignment). When the Extended Synch bit is set, SKP Ordered Sets must be scheduled and transmitted between FTSs and EIEOS as necessary to meet the definitions in Section 4.2.7. The last FTS Ordered set of the FTS sequence, if any (no FTS Ordered Sets are sent if N_FTS is equal to zero), is followed by a final EIEOS that will help the receiver acquire Block alignment. Implementations are permitted to send two EIEOS back to back following the last FTS Ordered Set if the N_FTS is a multiple of 32. The EIEOS resets the scrambler in both the Transmitter as well as the Receiver. Following the final EIEOS, an SDS Ordered Set is transmitted to help the receiver perform de-skew and to indicate the transition from Ordered

Sets to Data Stream. After the SDS Ordered Set is transmitted, a Data Block must be transmitted.



IMPLEMENTATION NOTE

Scrambling LFSR During FTS Transmission in 128b/130b Encoding

Since the scrambler is reset on the last EIEOS, and none of the ordered set in the FTS sequence is scrambled, it does not matter whether implementations choose to advance the scrambler or not during the time FTS is received.

Table 4-12: FTS for 8.0 GT/s and Above Data Rates

Symbol Number	Value
0	55h
1	47h
2	4Eh
3	C7h
4	CCh
5	C6h
6	C9h
7	25h
8	6Eh
9	ECh
10	88h
11	7Fh
12	80h
13	8Dh
14	8Bh
15	8Eh

N_FTS defines the number of FTSs that must be transmitted when transitioning from L0s to L0. At the 2.5 GT/s data rate, the value that can be requested by a component corresponds to a Symbol lock time of 16 ns (N_FTS set to 0b and one SKP Ordered Set) to ~4 µs (N_FTS set to 255), except when the Extended Synch bit is set, which requires the transmission of 4096 FTSs resulting in a bit lock time of 64 µs. For 8.0 GT/s and above data rates, when the Extended Synch bit is set, the transmitter is required to send 4096 FTS Ordered Set Blocks. Note that the N_FTS value reported by a component may change; for example, due to software modifying the value in the Common Clock Configuration bit (Section 7.8.7).

If the N_FTS period of time expires before the Receiver obtains bit lock, Symbol lock or Block alignment, and Lane-to-Lane de-skew on all Lanes of the configured Link, the Receiver must transition to the Recovery state. This sequence is detailed in the LTSSM in Section 4.2.5.

4.2.4.6. Start of Data Stream Ordered Set

The Start of Data Stream (SDS) Ordered Set, described in Table 4-13, is defined only for 128b/130b encoding. It is transmitted in the Configuration.Idle, Recovery.Idle, and Tx_L0s.FTS LTSSM states to define the transition from Ordered Set Blocks to a Data Stream, and Loopback Masters are permitted to transmit it as described in Section 4.2.2.5. It must not be transmitted at any other time. While not in the Loopback state, the Block following an SDS Ordered Set must be a Data Block, and the first Symbol of that Data Block is the first Symbol of the Data Stream.

Table 4-13: SDS Ordered Set (for 8.0 GT/s and Above Data Rate)

Symbol Number	Value	Description
0	E1	SDS Ordered Set Identifier
1-15	55h	Body of SDS Ordered Set

4.2.4.7. Link Error Recovery

- Link Errors, when operating with 8b/10b encoding are:
 - 8b/10b decode errors, Framing Errors, loss of Symbol lock, Elasticity Buffer Overflow/Underflow, or loss of Lane-to-Lane de-skew.
 - 8b/10b decode errors must be checked and trigger a Receiver Error in specified LTSSM states (see Table 4-14), which is a reported error associated with the Port (see Section 6.2). Triggering a Receiver Error on any or all of Framing Error, Loss of Symbol Lock, Lane De-skew Error, and Elasticity Buffer Overflow/Underflow is optional.
- Link Errors, when operating with 128b/130b encoding, are:
 - Framing Errors, loss of Block Alignment, Elasticity Buffer Overflow/Underflow, or loss of Lane-to-Lane de-skew.
 - Framing errors must be checked and trigger a Receiver Error in the LTSSM states specified in Table 4-14. The Receiver Error is a reported error associated with the Port (see Section 6.2). Triggering a Receiver Error on any of all of loss of Block Alignment, Elasticity Buffer Overflow/Underflow, and loss of Lane-to-Lane de-skew is optional.
- On a configured Link, which is in L0, error recovery will at a minimum be managed in a Layer above the Physical Layer (as described in Section 3.5) by directing the Link to transition to Recovery.
 - Note: Link Errors may also result in the Physical Layer initiating a LTSSM state transition from L0 to Recovery.

- All LTSSM states other than L0 make progress[43] when Link Errors occur.
 - When operating with 8b/10b encoding, Link Errors that occur in LTSSM states other than L0 must not result in the Physical Layer initiating an LTSSM state transition.
 - When operating with 128b/130b encoding and not processing a Data Stream, Link Errors that occur in LTSSM states other than L0 must not result in the Physical Layer initiating an LTSSM state transition.
- When operating with 8b/10b encoding, if a Lane detects an implementation specific number of 8b/10b errors, Symbol lock must be verified or re-established as soon as possible.[44]

4.2.4.8. Reset

Reset is described from a system point of view in Section 6.6.

4.2.4.8.1. Fundamental Reset

- Fundamental Reset applies only when Main power is present.
- Fundamental Reset does not apply when no power or Aux power is present.

When Fundamental Reset is asserted:

- The Receiver terminations are required to meet $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}POS}$ and $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}NEG}$ (see Table 4-24).
- The Transmitter is required only to meet $I_{TX\text{-}SHORT}$ (see Table 4-18).
- The Transmitter holds a constant DC common mode voltage.[45]

When Fundamental Reset is de-asserted:

- The Port LTSSM (see Section 4.2.5) is initialized (see Section 6.6.1 for additional requirements).

4.2.4.8.2. Hot Reset

Hot Reset is a protocol reset defined in Section 4.2.5.11.

[43] In this context, progress is defined as the LTSSM not remaining indefinitely in one state with the possible exception of Detect, Hot Reset, or Disabled.

[44] The method to verify and re-establish Symbol lock is implementation specific.

[45] The common mode being driven is not required to meet the Absolute Delta Between DC Common Mode during L0 and Electrical Idle ($V_{TX\text{-}CM\text{-}DC\text{-}ACTIVE\text{-}IDLE\text{-}DELTA}$) specification (see Table 4-18).

4.2.4.9. Link Data Rate Negotiation

All devices are required to start Link initialization using a 2.5 GT/s data rate on each Lane. A field in the training sequence Ordered Set (see Section 4.2.4) is used to advertise all supported data rates. The Link trains to L0 initially in 2.5 GT/s data rate after which a data rate change occurs by going through the Recovery state.

4.2.4.10. Link Width and Lane Sequence Negotiation

PCI Express Links must consist of 1, 2, 4, 8, 12, 16, or 32 Lanes in parallel, referred to as x1, x2, x4, x8, x12, x16, and x32 Links, respectively. All Lanes within a Link must simultaneously transmit data based on the same frequency with a skew between Lanes not to exceed $L_{TX-SKEW}$ (Table 4-24). The negotiation process is described as a sequence of steps.

The negotiation establishes values for Link number and Lane number for each Lane that is part of a valid Link; each Lane that is not part of a valid Link exits the negotiation to become a separate Link or remains in Electrical Idle.

During Link width and Lane number negotiation, the two communicating Ports must accommodate the maximum allowed Lane-to-Lane skew as specified by $L_{RX-SKEW}$ in Table 4-24.

Optional Link negotiation behaviors include Lane reversal, variable width Links, splitting of Ports into multiple Links and the configuration of a crosslink.

Other specifications may impose other rules and restrictions that must be comprehended by components compliant to those other specifications; it is the intent of this specification to comprehend interoperability for a broad range of component capabilities.

4.2.4.10.1. Required and Optional Port Behavior

- The ability for a xN Port to form a xN Link as well as a x1 Link (where N can be 32, 16, 12, 8, 4, 2, and 1) is required.
 - Designers must connect Ports between two different components in a way that allows those components to meet the above requirement. If the Ports between components are connected in ways that are not consistent with intended usage as defined by the component's Port descriptions/data sheets, behavior is undefined.
- The ability for a xN Port to form any Link width between N and 1 is optional.
 - An example of this behavior includes a x16 Port which can only configure into only one Link, but the width of the Link can be configured to be x12, x8, x4, x2 as well as the required widths of x16 and x1.
- The ability to split a Port into two or more Links is optional.
 - An example of this behavior would be a x16 Port that may be able to configure two x8 Links, four x4 Links, or 16 x1 Links.

- Support for Lane reversal is optional.
 - If implemented, Lane reversal must be done for both the Transmitter and Receiver of a given Port for a multi-Lane Link.
 - An example of Lane reversal consists of Lane 0 of an Upstream Port attached to Lane N-1 of a Downstream Port where either the Downstream or Upstream device may reverse the Lane order to configure a xN Link.

Support for formation of a crosslink is optional. In this context, a Downstream Port connected to a Downstream Port or an Upstream Port connected to an Upstream Port is a crosslink.

Current and future electromechanical and/or form factor specifications may require the implementation of some optional features listed above. Component designers must read the specifications for the systems that the component(s) they are designing will used in to ensure compliance to those specifications.

4.2.4.11. Lane-to-Lane De-skew

The Receiver must compensate for the allowable skew between all Lanes within a multi-Lane Link (see Table 4-18 and Table 4-24) before delivering the data and control to the Data Link Layer.

When using 8b/10b encoding, an unambiguous Lane-to-Lane de-skew mechanism is the COM Symbol of Ordered Sets received during training sequence or SKP Ordered Sets, since Ordered Sets are transmitted simultaneously on all Lanes of a configured Link.

When using 128b/130b encoding, an unambiguous Lane-to-Lane de-skew mechanism may use one or more of the following:

- A received SDS Ordered Set
- A received Electrical Idle Exit Ordered Set
- A received SKP Ordered Set

Other de-skew mechanisms may also be employed. Lane-to-Lane de-skew must be performed during Configuration, Recovery, and L0s in the LTSSM.

4.2.4.12. Lane vs. Link Training

The Link initialization process builds unassociated Lanes of a Port into associated Lanes that form a Link. For Lanes to configure properly into a desired Link, the TS1 and TS2 Ordered Sets must have the appropriate fields (Symbol 3, 4, and 5) set to the same values on all Lanes.

Links are formed at the conclusion of Configuration.

- If the optional behavior of a Port being able to configure multiple Links is employed, the following observations can be made:
 - A separate LTSSM is needed for each separate Link that is desired to be configured by any given Port.
 - The LTSSM Rules are written for configuring one Link. The decision to configure Links in a serial fashion or parallel is implementation specific.

4.2.5. Link Training and Status State Machine (LTSSM) Descriptions

The LTSSM states are illustrated in Figure 4-22. These states are described in following sections.

All timeout values specified for the Link Training and Status state machine (LTSSM) are minus 0 seconds and plus 50% unless explicitly stated otherwise. All timeout values must be set to the specified values after Fundamental Reset. All counter values must be set to the specified values after Fundamental Reset.

4.2.5.1. Detect

The purpose of this state is to detect when a far end termination is present. This state can be entered at any time if directed.

4.2.5.2. Polling

The Port transmits training Ordered Sets and responds to the received training Ordered Sets. In this state, bit lock and Symbol lock are established and Lane polarity is configured.

The polling state includes Polling.Compliance (see Section 4.2.6.2.2). This state is intended for use with test equipment used to assess if the Transmitter and the interconnect present in the device under test setup is compliant with the voltage and timing specifications in Table 4-18 and Table 4-24.

The Polling.Compliance state also includes a simplified inter-operability testing scheme that is intended to be performed using a wide array of test and measurement equipment (i.e., pattern generator, oscilloscope, BERT, etc.). This portion of the Polling.Compliance state is logically entered by at least one component asserting the Compliance Receive bit (bit 4 in Symbol 5 of TS1) while not asserting the Loopback bit (bit 2 in Symbol 5 of TS1) upon entering Polling.Active. The ability to set the Compliance Receive bit is implementation specific. A provision for changing data rates to that indicated by the highest common transmitted and received Data Rate Identifiers (Symbol 4 of TS1) is also included to make this behavior scalable to various data rates.



IMPLEMENTATION NOTE

Use of Polling.Compliance

Polling.Compliance is intended for a compliance test environment and not entered during normal operation and cannot be disabled for any reason. Polling.Compliance is entered based on the physical system environment or configuration register access mechanism as described in Section 4.2.6.2.1. Any other mechanism that causes a Transmitter to output the compliance pattern is implementation specific and is beyond the scope of this specification.

4.2.5.3. Configuration

In Configuration, both the Transmitter and Receiver are sending and receiving data at the negotiated data rate. The Lanes of a Port configure into a Link through a width and Lane negotiation sequence. Also, Lane-to-Lane de-skew must occur, scrambling can be disabled if permitted, the N_FTS is set, and the Disable or Loopback states can be entered.

4.2.5.4. Recovery

In Recovery, both the Transmitter and Receiver are sending and receiving data using the configured Link and Lane number as well as the previously supported data rate(s). Recovery allows a configured Link to change the data rate of operation if desired, re-establish bit lock, Symbol lock or Block alignment, and Lane-to-Lane de-skew. Recovery is also used to set a new N_FTS value and enter the Loopback, Disabled, Hot Reset, and Configuration states.

4.2.5.5. L0

L0 is the normal operational state where data and control packets can be transmitted and received. All power management states are entered from this state.

4.2.5.6. L0s

L0s is intended as a power savings state. When operating with separate reference clocks with independent Spread Spectrum Clocking (SSC) (see Section 4.2.7), L0s is not supported and must not be advertised in the capability registers. See Section 4.3.7.3 for a definition of SSC.

Comment [VS119]: ECN: SRIS

L0s allows a Link to quickly enter and recover from a power conservation state without going through Recovery.

The entry to L0s occurs after receiving an EIOS.

The exit from L0s to L0 must re-establish bit lock, Symbol lock or Block alignment, and Lane-to-Lane de-skew.

A Transmitter and Receiver Lane pair on a Port are not required to both be in L0s simultaneously.

4.2.5.7. L1

L1 is intended as a power savings state.

The L1 state allows an additional power savings over L0s at the cost of additional resume latency.

The entry to L1 occurs after being directed by the Data Link Layer and receiving an EIOS.

4.2.5.8. L2

Power can be aggressively conserved in L2. Most of the Transmitter and Receiver may be shut off.[46] Main power and clocks are not guaranteed, but Aux[47] power is available.

When Beacon support is required by the associated system or form factor specification, an Upstream Port that supports the wakeup capability must be able to send; and a Downstream Port must be able to receive; a wakeup signal referred to as a Beacon.

The entry to L2 occurs after being directed by the Data Link Layer and receiving an EIOS.

4.2.5.9. Disabled

The intent of the Disabled state is to allow a configured Link to be disabled until directed or Electrical Idle is exited (i.e., due to a hot removal and insertion) after entering Disabled.

Disabled uses bit 1 (Disable Link) in the Training Control field (see Table 4-5 and Table 4-6) which is sent within the TS1 and TS2 Ordered Sets.

A Link can enter Disabled if directed by a higher Layer. A Link can also reach the Disabled state by receiving two consecutive TS1 Ordered Sets with the Disable Link bit asserted (see Sections 4.2.6.3.1 and 4.2.6.4.5).

4.2.5.10. Loopback

Loopback is intended for test and fault isolation use. Only the entry and exit behavior is specified, all other details are implementation specific. Loopback can operate on either a per Lane or configured Link basis.

A loopback master is the component requesting Loopback.

A loopback slave is the component looping back the data.

Loopback uses bit 2 (Loopback) in the Training Control field (see Table 4-5 and Table 4-6) which is sent within the TS1 and TS2 Ordered Sets.

The entry mechanism for a loopback master is device specific.

The loopback slave device enters Loopback whenever two consecutive TS1 Ordered Sets are received with the Loopback bit set.

[46] The exception is the Receiver termination, which must remain in a low impedance state.

[47] In this context, “Aux” power means a power source which can be used to drive the Beacon circuitry.



IMPLEMENTATION NOTE

Use of Loopback

Once in the Loopback state, the master can send any pattern of Symbols as long as the encoding rules are followed. Once in Loopback, the concept of data scrambling is no longer relevant; what is sent out is looped back. The mechanism(s) and/or interface(s) utilized by the Data Link Layer to notify the Physical Layer to enter the Loopback state is component implementation specific and beyond the scope of this specification.

4.2.5.11. Hot Reset

Hot Reset uses bit 0 (Hot Reset) in the Training Control field (see Table 4-5 and Table 4-6) within the TS1 and TS2 Ordered Sets.

A Link can enter Hot Reset if directed by a higher Layer. A Link can also reach the Hot Reset state by receiving two consecutive TS1 Ordered Sets with the Hot Reset bit asserted (see Section 4.2.6.11).

4.2.6. Link Training and Status State Rules

Various Link status bits are monitored through software with the exception of LinkUp which is monitored by the Data Link Layer. Table 4-14 describes how the Link status bits must be handled throughout the LTSSM (for more information, see Section 3.2 for LinkUp; Section 7.8.8 for Link Speed, Link Width, and Link Training; Section 6.2 for Receiver Error; and Section 6.7 for In-Band Presence). A Receiver may also optionally report an 8b/10b Error in the Lane Error Status register when operating in 8b/10b encoding, when allowed to report the error as a Receiver Error in Table 4-14.



IMPLEMENTATION NOTE

Receiver Errors During Configuration and Recovery States

Allowing Receiver Errors to be set while in Configuration or Recovery is intended to allow implementations to report Link Errors that occur while processing packets in those states. For example, if the LTSSM transitions from L0 to Recovery while a TLP is being received, a Link Error that occurs after the LTSSM transition can be reported.

Table 4-14: Link Status Mapped to the LTSSM

LTSSM State	Link Width	Link Speed	LinkUp	Link Training	Receiver Error	In-Band Presence[48]
Detect	Undefined	Undefined	0b	0b	No action	0b
Polling	Undefined	Set to 2.5 GT/s on entry from Detect. Link speed may change on entry to Polling.Compliance.	0b	0b	No action	1b
Configuration	Set	No action	0b/1b[49]	1b	Set on 8b/10b Error. Optional: Set on Link Error when using 128b/130b encoding.	1b
Recovery	No action	Set to new speed when speed changes	1b	1b	Optionally set on Link Error.	1b
L0	No action	No action	1b	0b	Set on Link Error.	1b
L0s	No action	No action	1b	0b	No action	1b
L1	No action	No action	1b	0b	No action	1b
L2	No action	No action	1b	0b	No action	1b
Disabled	Undefined	Undefined	0b	0b	Optional: Set on 8b/10b Error	1b
Loopback	No action	Link speed may change on entry to Loopback from Configuration.	0b	0b	No action	1b
Hot Reset	No action	No action	0b	0b	Optional: Set on 8b/10b Error	1b

The state machine rules for configuring and operating a PCI Express Link are defined in the following sections.

[48] In-band refers to the fact that no sideband signals are used to calculate the presence of a powered up device on the other end of a Link.

[49] LinkUp will always be 0 if coming into Configuration via Detect -> Polling -> Configuration and LinkUp will always be 1 if coming into Configuration from any other state.

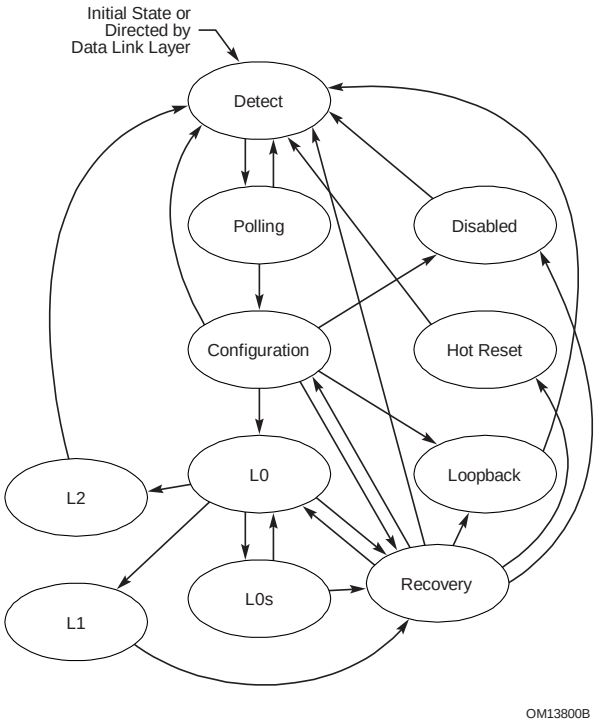


Figure 4-22: Main State Diagram for Link Training and Status State Machine

4.2.6.1. Detect

The Detect substate machine is shown in Figure 4-23.

4.2.6.1.1. Detect.Quiet

- Transmitter is in an Electrical Idle state.
 - The DC common mode voltage is not required to be within specification.
- 2.5 GT/s data rate is selected as the frequency of operation. If the frequency of operation was not 2.5 GT/s data rate on entry to this substate, the LTSSM must stay in this substate for at least 1 ms, during which the frequency of operation must be changed to the 2.5 GT/s data rate.
 - Note: This does not affect the advertised data rate in the TS1 and TS2 Ordered Sets.
- LinkUp = 0b (status is cleared).
- The Equalization Phase 1 Successful, Equalization Phase 2 Successful, Equalization Phase 3 Successful, and Equalization Complete bits of the Link Status 2 register are all set to 0b.
- The directed_speed_change variable is reset to 0b. The upconfigure_capable variable is reset to 0b. The idle_to_rlock_transitioned variable is reset to 00h. The select_deemphasis variable

must be set to either 0b or 1b based on platform specific needs for an Upstream Port and identical to the Selectable Preset/De-emphasis field in the Link Control 2 register for a Downstream Port. The equalization_done_8GT_data_rate variable is reset to 0b.

- Note that since these variables are defined with the 2.0 specification, pre-2.0 devices would not implement these variables and will always take the path as if the directed_speed_change and upconfigure_capable variables are constantly reset to 0b and the idle_to_rlock_transitioned variable is constantly set to FFh.

- ❑ The next state is Detect.Active after a 12 ms timeout or if Electrical Idle is broken on any Lane.

4.2.6.1.2. Detect.Active

- ❑ The Transmitter performs a Receiver Detection sequence on all un-configured Lanes that can form one or more Links (see Section 4.3.5.6 for more information).
- ❑ Next state is Polling if a Receiver is detected on all unconfigured Lanes.
- ❑ Next state is Detect.Quiet if a Receiver is not detected on any Lane.
- ❑ If at least one but not all un-configured Lanes detect a Receiver, then:

1. Wait for 12 ms.
2. The Transmitter performs a Receiver Detection sequence on all un-configured Lanes that can form one or more Links (see Section 4.3.5.6 for more information),
 - The next state is Polling if exactly the same Lanes detect a Receiver as the first Receiver Detection sequence.
 - Lanes that did not detect a Receiver must:

 i) Be associated with a new LTSSM if this optional feature is supported.

 or

 ii) All Lanes that cannot be associated with an optional new LTSSM must transition to Electrical Idle.[50]
 - These Lanes must be re-associated with the LTSSM immediately after the LTSSM in progress transitions back to Detect.
 - An EIOS does not need to be sent before transitioning to Electrical Idle.
 - Otherwise, the next state is Detect.Quiet.

[50] The common mode being driven is not required to meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$) specification (see Table 4-18).

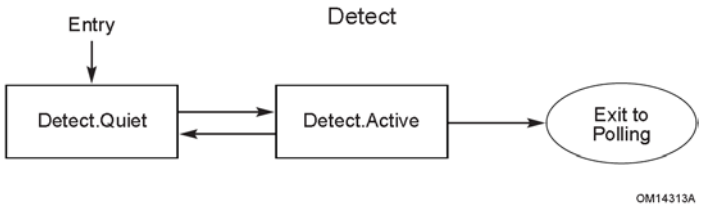


Figure 4-23: Detect Substate Machine

4.2.6.2. Polling

The Polling substate machine is shown in Figure 4-24.

4.2.6.2.1. Polling.Active

- Transmitter sends TS1 Ordered Sets with Lane and Link numbers set to PAD on all Lanes that detected a Receiver during Detect.
 - The Data Rate Identifier Symbol of the TS1 Ordered Sets must advertise all data rates that the Port supports, including those that it does not intend to use.
 - The Transmitter must wait for its TX common mode to settle before exiting from Electrical Idle and transmitting the TS1 Ordered Sets.
 - The Transmitter must drive patterns in the default voltage level of the Transmit Margin field within 192 ns from entry to this state. This transmit voltage level will remain in effect until Polling.Compliance or Recovery.RcvrLock is entered.
- Next state is Polling.Compliance if the Enter Compliance bit (bit 4) in the Link Control 2 register is 1b. If the Enter Compliance bit was set prior to entry to Polling.Active, the transition to Polling.Compliance must be immediate without sending any TS1 Ordered Sets.
- Next state is Polling.Configuration after at least 1024 TS1 Ordered Sets were transmitted, and all Lanes that detected a Receiver during Detect receive eight consecutive training sequences (or their complement) satisfying any of the following conditions:
 - TS1 with Lane and Link numbers set to PAD and the Compliance Receive bit (bit 4 of Symbol 5) is 0b.
 - TS1 with Lane and Link numbers set to PAD and the Loopback bit (bit 2 of Symbol 5) is 1b.
 - TS2 with Lane and Link numbers set to PAD.

- Otherwise, after a 24 ms timeout the next state is:
 - Polling.Configuration if,

(i) Any Lane, which detected a Receiver during Detect, received eight consecutive training sequences (or their complement) satisfying any of the following conditions:

1. TS1 with Lane and Link numbers set to PAD and the Compliance Receive bit (bit 4 of Symbol 5) is 0b.
2. TS1 with Lane and Link numbers set to PAD and the Loopback bit (bit 2 of Symbol 5) is 1b.
3. TS2 with Lane and Link numbers set to PAD.

and a minimum of 1024 TS1 Ordered Sets are transmitted after receiving one TS1 or TS2 Ordered Set[51].

Comment [VS120]: Errata A6a (replacement for A6)

And

(ii) At least a predetermined set of Lanes that detected a Receiver during Detect have detected an exit from Electrical Idle at least once since entering Polling.Active.

- Note: This may prevent one or more bad Receivers or Transmitters from holding up a valid Link from being configured, and allow for additional training in Polling.Configuration. The exact set of predetermined Lanes is implementation specific. Note that up to the 1.1 specification this predetermined set was equal to the total set of Lanes that detected a Receiver.
- Note: Any Lane that receives eight consecutive TS1 or TS2 Ordered Sets should have detected an exit from Electrical Idle at least once since entering Polling.Active.

 - Else Polling.Compliance if either (a) or (b) is true:

(a) not all Lanes from the predetermined set of Lanes from (ii) above have detected an exit from Electrical Idle since entering Polling.Active.

(b) any Lane that detected a Receiver during Detect received eight consecutive TS1 Ordered Sets (or their complement) with the Lane and Link numbers set to PAD, the Compliance Receive bit (bit 4 of Symbol 5) is 1b, and the Loopback bit (bit 2 of Symbol 5) is 0b.

- Note: If a passive test load is applied on all Lanes then the device will go to Polling.Compliance.

 - Else Detect if the conditions to transition to Polling.Configuration or Polling.Compliance are not met

[51] Earlier versions of this specification required transmission of 1024 TS1 Ordered Sets after receiving one TS1 Ordered Set. This behavior is still permitted but the implementation will be more robust if it follows the behavior of transmitting 1024 TS1 Ordered Sets after receiving one TS1 or TS2 Ordered Set.

4.2.6.2.2. Polling.Compliance

- The Transmit Margin field of the Link Control 2 register is sampled on entry to this substate and becomes effective on the transmit package pins within 192 ns of entry to this substate and remain effective through the time the LTSSM is in this substate.
- The data rate and de-emphasis level for transmitting the compliance pattern are determined on the transition from Polling.Active to Polling.Compliance using the following algorithm.
 - If the Port is capable of transmitting at the 2.5 GT/s data rate only, the data rate for transmitting the compliance pattern is 2.5 GT/s and the de-emphasis level is -3.5 dB.
 - Else if the Port entered Polling.Compliance due to detecting eight consecutive TS1 Ordered Sets in Polling.Active with the Compliance Receive bit (bit 4 of Symbol 5) asserted and the Loopback bit (bit 2 of Symbol 5) deasserted then the data rate for transmission is that indicated by the highest common transmitted and received Data Rate Identifiers (Symbol 4 of the TS1 sequence) advertised on the eight consecutive TS1 Ordered Sets received on any Lane that detected a Receiver during Detect. The select_deemphasis variable must be set equal to the Selectable De-emphasis bit (Symbol 4 bit 6) in the eight consecutive TS1 Ordered Sets it received in Polling.Active substate. If the common data rate is 8.0 GT/s, the select_preset variable on each Lane is set to the transmitter preset value advertised in Symbol 6 of the eight consecutive TS1 Ordered Sets and this value must be used by the transmitter (use of the receiver preset value advertised in those TS1 Ordered Sets is optional). If the common data rate is 8.0 GT/s, any Lanes that did not receive eight consecutive TS1 Ordered Sets with Transmitter Preset information can use any supported Transmitter Preset in an implementation specific manner.
 - Else if the Enter Compliance bit in the Link Control 2 register is 1b, the data rate for transmitting the compliance pattern is defined by the Target Link Speed field in the Link Control 2 register. The select_deemphasis variable is Set when the Compliance Preset/De-emphasis field in the Link Control 2 register equals 0001b if the data rate will be 5.0 GT/s. If the data rate will be 8.0 GT/s, the select_preset variable on each Lane is set to, and the transmitter must operate with, the preset value provided in the Compliance Preset/De-emphasis Value (bits 15:12) in the Link Control 2 register provided the value is not a Reserved encoding.
 - Else the data rate, preset, and de-emphasis level settings are defined as follows based on the component's maximum supported data rate and the number of times Polling.Compliance has been entered with this entry criteria:

 Setting #1: Data Rate = 2.5 GT/s, De-emphasis Level = -3.5 dB

 Setting #2: Data Rate = 5.0 GT/s, De-emphasis Level = -3.5 dB

 Setting #3: Data Rate = 5.0 GT/s, De-emphasis Level = -6 dB

 Setting #4: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0000b defined in Section 4.2.3.2

 Setting #5: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0001b defined in Section 4.2.3.2

Setting #6: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0010b defined in Section 4.2.3.2

Setting #7: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0011b defined in Section 4.2.3.2

Setting #8: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0100b defined in Section 4.2.3.2

Setting #9: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0101b defined in Section 4.2.3.2

Setting #10: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0110b defined in Section 4.2.3.2

Setting #11: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0111b defined in Section 4.2.3.2

Setting #12: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 1000b defined in Section 4.2.3.2

Setting #13: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 1001b defined in Section 4.2.3.2

Setting #14: Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 1010b defined in Section 4.2.3.2

Subsequent entries to Polling.Compliance repeat the above sequence. For example, the state sequence which causes a Port to transmit the Compliance pattern at a data rate of 5.0 GT/s and a de-emphasis level of -6 dB is: Polling.Active, Polling.Compliance (2.5 GT/s and -3.5 dB), Polling.Active, Polling.Compliance (5.0 GT/s and -3.5 dB), Polling.Active, Polling.Compliance (5.0 GT/s and -6 dB).

The sequence must be set to Setting #1 in the Polling.Configuration state if the Port's Receivers do not meet the Z_{RX-DC} specification for 2.5 GT/s when they are operating at 8.0 GT/s or higher data rates (see Table 4-24). All Ports are permitted to set the sequence to Setting #1 in the Polling.Configuration state.

Comment [VS121]: ECR: 8.0 GT/s Receiver Impedance

IMPLEMENTATION NOTE

Compliance Load Board Usage to Generate Compliance Patterns

It is envisioned that the compliance load (base) board may send a 100 MHz signal for about 1 ms on one leg of a differential pair at 350 mV peak-to-peak on any Lane to cycle the device to the desired speed and de-emphasis level. The device under test is required, based on its maximum supported data rate, to cycle through the following settings in order, for each entry to Polling.Compliance from Polling.Active, starting with the first setting on the first entry to Polling.Compliance after the Fundamental Reset:

- Data Rate = 2.5 GT/s, De-emphasis Level = -3.5 dB
- Data Rate = 5.0 GT/s, De-emphasis Level = -3.5 dB
- Data Rate = 5.0 GT/s, De-emphasis Level = -6 dB
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0000b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0001b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0010b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0011b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0100b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0101b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0110b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 0111b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 1000b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 1001b defined in Section 4.2.3.2
- Data Rate = 8.0 GT/s, with Transmitter Preset Encoding 1010b defined in Section 4.2.3.2

- If the compliance pattern data rate is not 2.5 GT/s and any TS1 Ordered Sets were transmitted in Polling.Active prior to entering Polling.Compliance, the Transmitter sends either one EIOS or two consecutive EIOSs prior to entering Electrical Idle. If the compliance pattern data rate is not 2.5 GT/s and TS1 Ordered Sets were not transmitted in Polling.Active prior to entering Polling.Compliance, the Transmitter must enter Electrical Idle without transmitting any EIOSs. During the period of Electrical Idle, the data rate is changed to the new speed and stabilized. If the frequency of operation will be 5.0 GT/s, the de-emphasis/preset level must be set to -3.5 dB if the select_deemphasis variable is 1b else it must be set to -6 dB. If the frequency of operation will be 8.0 GT/s, the transmitter preset value must be set to the value in the select_preset variable. The period of Electrical Idle is greater than 1 ms but it is not to exceed 2 ms.

- Behavior during Polling Compliance after the data rate and de-emphasis/preset level are determined must follow the following rules:
 - If the Port entered Polling.Compliance due to detecting eight consecutive TS1 Ordered Sets in Polling.Active with the Compliance Receive bit (bit 4 of Symbol 5) asserted and the Loopback bit (bit 2 of Symbol 5) deasserted or both the Enter Compliance bit and the Enter Modified Compliance bit in the Link Control 2 register are set to 1b then the Transmitter sends out the Modified Compliance Pattern (see Section 4.2.9) at the above determined data rate with the error status Symbol set to all 0's on all Lanes that detected a Receiver during Detect.
 - If the data rate is 2.5 GT/s or 5.0 GT/s, a particular Lane's Receiver independently signifies a successful lock to the incoming Modified Compliance Pattern by looking for any one occurrence of the Modified Compliance Pattern and then setting the Pattern Lock bit (bit 8 of the 8 bit error status Symbol) in the same Lane of its own transmitted Modified Compliance Pattern.
 - The error status Symbols are not to be used for the lock process since they are undefined at any given moment.
 - An occurrence is defined above as the following sequence of 8b/10b Symbols; K28.5, D21.5, K28.5, and D10.2 or the complement of each of the individual Symbols.
 - The device under test must set the Pattern Lock bit of the Modified Compliance Pattern it transmits at the Transmitter package pin(s) after successfully locking to the incoming Modified Compliance Pattern within 1 ms of receiving the Modified Compliance Pattern at its Receiver package pin(s).
 - If the data rate is 8.0 GT/s: The Error_Status field is set to 00h on entry to this substate. Each Lane sets the Pattern Lock bit independently when it achieves Block Alignment as described in Section 4.2.2.2.1. After Pattern Lock is achieved, Symbols received in Data Blocks are compared to the Idle data Symbol (00h) and each mismatched Symbol causes the Receiver Error Count field to be incremented by 1. The Receiver Error Count saturates at 127 (further mismatched Symbols do not change the Receiver Error Count). The Pattern Lock and Receiver Error Count information for each Lane is transmitted as part of the SKP Ordered Sets transmitted in that Lane's Modified Compliance Pattern. See Section 4.2.7 for more information. The device under test must set the Pattern Lock bit in the SKP Ordered Set it transmits within 4 ms of receiving the Modified Compliance Pattern at its Receiver package pin(s).

 The scrambling requirements defined in Section 4.2.2.4 are applied to the received Modified Compliance Pattern. For example, the scrambling LFSR seed is set per Lane, an EIEOS initializes the LFSR and SKP Ordered Sets do not advance the LFSR.



IMPLEMENTATION NOTE

Handling Bit Slip and Block Alignment

Devices should ensure that their Receivers have stabilized before attempting to obtain Block alignment and signaling Pattern Lock. For example, if an implementation expects to see bit slips in the initial few bits, it should wait for that time to be over before settling on a Block Alignment. Devices may also want to revalidate their Block alignment prior to setting the Pattern Lock bit.

- If the data rate is 2.5 GT/s or 5.0 GT/s, once a particular Lane indicates it has locked to the incoming Modified Compliance Pattern the Receiver Error Count for that particular Lane is incremented every time a Receiver error occurs.
 - The error status Symbol uses the lower 7 bits as the Receiver Error Count field and this field will remain stuck at all 1's if the count reaches 127.
 - The Receiver must not make any assumption about the 10-bit patterns it will receive when in this substate if 8b/10b encoding is used.
- If the Enter Compliance bit in the Link Control 2 register is 0b, the next state is Detect if directed
- Else if the Enter Compliance bit was set to 1b on entry to Polling.Compliance, next state is Polling.Active if any of the following conditions apply:
 - The Enter Compliance bit in the Link Control 2 register has changed to 0b
 - The Port is an Upstream Port and an EIOS is received on any Lane. The Enter Compliance bit is reset to 0b when this condition is true.

 If the Transmitter was transmitting at a data rate other than 2.5 GT/s, or the Enter Compliance bit in the Link Control 2 register was set to 1b during entry to Polling.Compliance, the Transmitter sends eight consecutive EIOS and enters Electrical Idle prior to transitioning to Polling.Active. During the period of Electrical Idle, the data rate is changed to 2.5 GT/s and stabilized and the de-emphasis level is set to -3.5 dB. The period of Electrical Idle is greater than 1 ms but must not exceed 2 ms.
- Note: Sending multiple EIOS provides enough robustness such that the other Port detects at least one EIOS and exits Polling.Compliance substate when the configuration register mechanism was used for entry.

o Else (i.e., entry to Polling.Compliance is not due to the Compliance Receive bit assertion with Loopback bit deassertion in the received consecutive TS Ordered Sets and not due to the Enter Compliance bit and the Enter Modified Compliance bit in the Link Control 2 register set to 1b)

(a) Transmitter sends out the compliance pattern on all Lanes that detected a Receiver during Detect at the data rate and de-emphasis/preset level determined above.

(b) Next state is Polling.Active if any of the following three conditions are true:

1. Electrical Idle exit has been detected at the Receiver of any Lane that detected a Receiver during Detect, and the Enter Compliance bit in the Link Control 2 register is 0b.

Note: the "any Lane" condition supports the usage model described in the "Compliance Load Board Usage to Generate Compliance Patterns" Implementation Note earlier in this section.

2. The Enter Compliance bit in the Link Control 2 register has changed to 0b (from 1b) since entering Polling.Compliance.

3. The Port is an Upstream Port, the Enter Compliance bit in the Link Control 2 register is set to 1b and an EIOS has been detected on any Lane. The Enter Compliance bit is reset to 0b when this condition is true.

If the Transmitter is transmitting at a data rate other than 2.5 GT/s, or the Enter Compliance bit in the Link Control 2 register was set to 1b during entry to Polling.Compliance, the Transmitter sends eight consecutive EIOSs and enters Electrical Idle prior to transitioning to Polling.Active. During the period of Electrical Idle, the data rate is changed to 2.5 GT/s and stabilized. The period of Electrical Idle is greater than 1 ms but must not exceed 2 ms.

Note: Sending multiple EIOSs provides enough robustness such that the other Port detects at least one EIOS and exits Polling.Compliance substate when the configuration register mechanism was used for entry.

4.2.6.2.3. Polling.Configuration

- ❑ Receiver must invert polarity if necessary (see Section 4.2.4.4).
- ❑ The Transmit Margin field of the Link Control 2 register must be reset to 000b on entry to this substate.
- ❑ The Transmitter's Polling.Compliance sequence setting is updated, if required, as described in Section 4.2.6.2.2.
- ❑ Transmitter sends TS2 Ordered Sets with Link and Lane numbers set to PAD on all Lanes that detected a Receiver during Detect.
 - o The Data Rate Identifier Symbol of the TS2 Ordered Sets must advertise all data rates that the Port supports, including those that it does not intend to use.
- ❑ The next state is Configuration after eight consecutive TS2 Ordered Sets, with Link and Lane numbers set to PAD, are received on any Lanes that detected a Receiver during Detect, and 16 TS2 Ordered Sets are transmitted after receiving one TS2 Ordered Set.
- ❑ Otherwise, next state is Detect after a 48 ms timeout.

Comment [VS122]: ECR: 8.0 GT/s Receiver Impedance

4.2.6.2.4. Polling.Speed

This state is unreachable given that the Link comes up to L0 in 2.5 GT/s data rate only and changes speed by entering Recovery.

IMPLEMENTATION NOTE

Support for Higher Data Rates than 2.5 GT/s

A Link will initially train to the L0 state at the 2.5 GT/s data rate even if both sides are capable of operating at a data rate greater than 2.5 GT/s. Supported higher data rates are advertised in the TS Ordered Sets. The other side's speed capability is registered during the Configuration.Complete substate. Based on the highest supported common data rate, either side can initiate a change in speed from the L0 state by transitioning to Recovery.

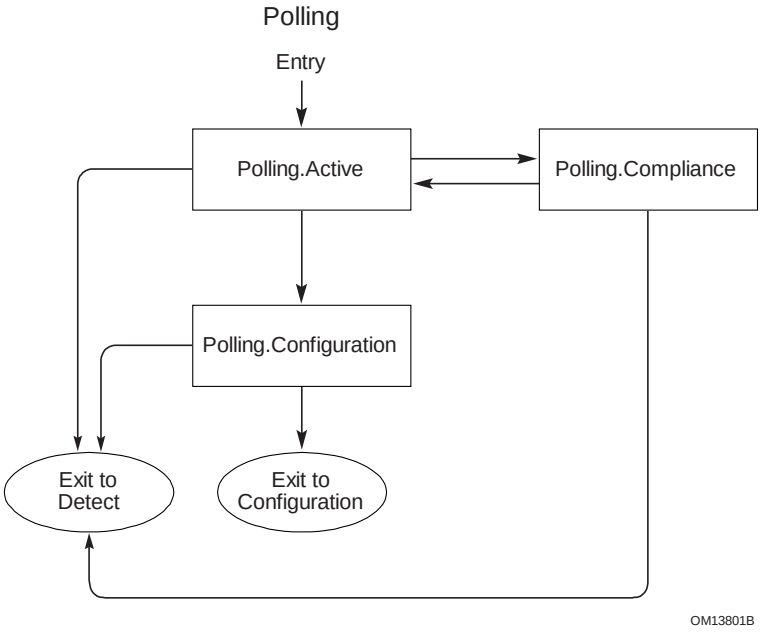


Figure 4-24: Polling Substate Machine

4.2.6.3. Configuration

The Configuration substate machine is shown in Figure 4-25.

4.2.6.3.1. Configuration.Linkwidth.Start

4.2.6.3.1.1. Downstream Lanes

- Next state is Disabled if directed.
 - Note: "if directed" applies to a Downstream Port that is instructed by a higher Layer to assert the Disable Link bit (TS1 and TS2) on all Lanes that detected a Receiver during Detect.
- Next state is Loopback if directed to this state, and the Transmitter is capable of being a loopback master, which is determined by implementation specific means.
 - Note: "if directed" applies to a Port that is instructed by a higher Layer to assert the Loopback bit (TS1 and TS2) on all Lanes that detected a Receiver during Detect.
- In the optional case where a crosslink is supported, the next state is Disabled after all Lanes that are transmitting TS1 Ordered Sets receive two consecutive TS1 Ordered Sets with the Disable Link bit asserted.
- Next state is Loopback after all Lanes that are transmitting TS1 Ordered Sets, that are also receiving Ordered Sets, receive the Loopback bit asserted in two consecutive TS1 Ordered Sets.

Note that the device receiving the Ordered Set with the Loopback bit set becomes the loopback slave.

- The Transmitter sends TS1 Ordered Sets with selected Link numbers and sets Lane numbers to PAD on all the active Downstream Lanes if LinkUp is 0b or if the LTSSM is not initiating upconfiguration of the Link width. In addition, if upconfigure_capable is set to 1b, and the LTSSM is not initiating upconfiguration of the Link width, the LTSSM sends TS1 Ordered Sets with the selected Link number and sets the Lane number to PAD on each inactive Lane after it detected an exit from Electrical Idle since entering Recovery and has subsequently received two consecutive TS1 Ordered Sets with the Link and Lane numbers each set to PAD while in this substate.
 - On transition to this substate from Polling, any Lane that detected a Receiver during Detect is considered an active Lane.
 - On transition to this substate from Recovery, any Lane that is part of the configured Link the previous time through Configuration.Complete is considered an active Lane.
 - The Data Rate Identifier Symbol of the TS1 Ordered Sets must advertise all data rates that the Port supports, including those that it does not intend to use.

- If LinkUp is 1b and the LTSSM is initiating upconfiguration of the Link width, initially it transmits TS1 Ordered Sets with both the Link and Lane numbers set to PAD on the current set of active Lanes; the inactive Lanes it intends to activate; and those Lanes where it detected an exit from Electrical Idle since entering Recovery and has received two consecutive TS1 Ordered Sets with the Link and Lane numbers each set to PAD. The LTSSM transmits TS1 Ordered Sets with the selected Link number and the Lane number set to PAD when each of the Lanes transmitting TS1 Ordered Sets receives two consecutive TS1 Ordered Sets with the Link and Lane numbers each set to PAD or 1 ms has expired since entering this substate.
 - After activating any inactive Lane, the Transmitter must wait for its TX common mode to settle before exiting from Electrical Idle and transmitting the TS1 Ordered Sets.
 - Link numbers are only permitted to be different for groups of Lanes capable of being a unique Link.
 - Note: An example of Link number assignments is a set of eight Downstream Lanes capable of negotiating to become one x8 Port when connected to one component or two x4 Ports when connected to two different components. The Downstream Lanes send out TS1 Ordered Sets with the Link number set to N on four Lanes and Link number set to N+1 on the other four Lanes. The Lane numbers are all set to PAD.
- If any Lanes first received at least one or more TS1 Ordered Sets with a Link and Lane number set to PAD, the next state is Configuration.Linkwidth.Accept immediately after any of those same Downstream Lanes receive two consecutive TS1 Ordered Sets with a non-PAD Link number that matches any of the transmitted Link numbers, and with a Lane number set to PAD.
 - If the crosslink configuration is not supported, the condition of first receiving a Link and Lane number set to PAD is always true.
- Else: Optionally, if LinkUp is 0b and if crosslinks are supported, then all Downstream Lanes that detected a Receiver during Detect must first transmit 16-32 TS1 Ordered Sets with a non PAD Link number and PAD Lane number and after this occurs if any Downstream Lanes receive two consecutive TS1 Ordered Sets with a Link number different than PAD and a Lane Number set to PAD, the Downstream Lanes are now designated as Upstream Lanes and a new random crosslink timeout is chosen (see $T_{crosslink}$ in Table 4-18). The next state is Configuration.Linkwidth.Start as Upstream Lanes.
 - Note: This supports the optional crosslink where both sides may try to act as a Downstream Port. This is resolved by making both Ports become Upstream and assigning a random timeout until one side of the Link becomes a Downstream Port and the other side remains an Upstream Port. This timeout must be random even when hooking up two of the same devices so as to eventually break any possible deadlock.
 - If crosslinks are supported, receiving a sequence of TS1 Ordered Sets with a Link number of PAD followed by a Link number of non-PAD that matches the transmitted Link number is only valid when not interrupted by the reception of a TS2 Ordered Set.



IMPLEMENTATION NOTE

Crosslink Initialization

In the case where the Downstream Lanes are connected to both Downstream Lanes (crosslink) and Upstream Lanes, the Port with the Downstream Lanes may continue with a single LTSSM as described in this section or optionally, split into multiple LTSSMs.

- The next state is Detect after a 24 ms timeout.

4.2.6.3.1.2. Upstream Lanes

- In the optional case where crosslinks are supported the next state is Disabled if directed.
 - Note: "if directed" only applies to an optional crosslink Port that is instructed by a higher Layer to assert the Disable Link bit (TS1 and TS2) on all Lanes that detected a Receiver during Detect.
- Next state is Loopback if directed to this state, and the Transmitter is capable of being a loopback master, which is determined by implementation specific means.
 - Note: "if directed" applies to a Port that is instructed by a higher Layer to assert the Loopback bit (TS1 and TS2) on all Lanes that detected a Receiver during Detect.
- Next state is Disabled after any Lanes that are transmitting TS1 Ordered Sets receive two consecutive TS1 Ordered Sets with the Disable Link bit asserted.
 - In the optional case where a crosslink is supported, the next state is Disabled only after all Lanes that are transmitting TS1 Ordered Sets, that are also receiving TS1 Ordered Sets, receive the Disable Link bit asserted in two consecutive TS1 Ordered Sets.
- Next state is Loopback after all Lanes that are transmitting TS1 Ordered Sets, that are also receiving TS1 Ordered Sets, receive the Loopback bit asserted in two consecutive TS1 Ordered Sets.
 - Note: The device receiving the Ordered Set with the Loopback bit set becomes the loopback slave.
- The Transmitter sends out TS1 Ordered Sets with Link numbers and Lane numbers set to PAD on all the active Upstream Lanes; the inactive Lanes it is initiating to upconfigure the Link width; and if upconfigure_capable is set to 1b, on each of the inactive Lanes where it detected an exit from Electrical Idle since entering Recovery and has subsequently received two consecutive TS1 Ordered Sets with Link and Lane numbers, each set to PAD, in this substate.
 - On transition to this substate from Polling, any Lane that detected a Receiver during Detect is considered an active Lane.
 - On transition to this substate from Recovery, any Lane that is part of the configured Link the previous time through Configuration.Complete is considered an active Lane.
 - On transition to this substate from Recovery, if the transition is not caused by LTSSM timeout, the Transmitter must set the Autonomous Change (Symbol 4 bit 6) to 1b in the

TS1 Ordered Sets that it sends while in the Configuration state if the Transmitter intends to change the Link width for autonomous reasons.

- The Data Rate Identifier Symbol of the TS1 Ordered Sets must advertise all data rates that the Port supports, including those that it does not intend to use.

- If any Lane receives two consecutive TS1 Ordered Sets with Link numbers that are different than PAD and Lane number set to PAD, a single Link number is selected and Lane number set to PAD are transmitted on all Lanes that both detected a Receiver and also received two consecutive TS1 Ordered Sets with Link numbers that are different than PAD and Lane number set to PAD. Any left over Lanes that detected a Receiver during Detect must transmit TS1 Ordered Sets with the Link and Lane number set to PAD. The next state is Configuration.Linkwidth.Accept:
 - If the LTSSM is initiating upconfiguration of the Link width, it waits until it receives two consecutive TS1 Ordered Sets with a non-PAD Link Number and a PAD Lane number on all the inactive Lanes it wants to activate, or, 1 ms after entry to this substate, it receives two consecutive TS1 Ordered Sets on any Lane with a non-PAD Link number and PAD Lane number, whichever occurs earlier, before transmitting TS1 Ordered Sets with selected Link number and Lane number set to PAD.
 - It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set or lost 128b/130b Block Alignment on a subset of the received Lanes; delay the evaluation listed above by an additional two, or more, TS1 Ordered Sets when using 8b/10b encoding, or by an additional 34, or more, TS1 Ordered Sets when using 128b/130b encoding, but must not exceed 1 ms, so as not to prematurely configure a smaller Link than possible.
 - After activating any inactive Lane, the Transmitter must wait for its TX common mode to settle before exiting Electrical Idle and transmitting the TS1 Ordered Sets.

- Optionally, if LinkUp is 0b and if crosslinks are supported, then all Upstream Lanes that detected a Receiver during Detect must first transmit 16–32 TS1 Ordered Sets with a PAD Link number and PAD Lane number and after this occurs and if any Upstream Lanes first receive two consecutive TS1 Ordered Sets with Link and Lane numbers set to PAD, then:
 - The Transmitter continues to send out TS1 Ordered Sets with Link numbers and Lane numbers set to PAD.
 - If any Lanes receive two consecutive TS1 Ordered Sets with Link numbers that are different than PAD and Lane number set to PAD, a single Link number is selected and Lane number set to PAD are transmitted on all Lanes that both detected a Receiver and also received two consecutive TS1 Ordered Sets with Link numbers that are different than PAD and Lane number set to PAD. Any left over Lanes that detected a Receiver during Detect must transmit TS1 Ordered Sets with the Link and Lane number set to PAD. The next state is Configuration.Linkwidth.Accept.
 - It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set or lost 128b/130b Block Alignment on a subset of the received Lanes; delay the evaluation listed above by an additional two, or more, TS1 Ordered Sets when using 8b/10b encoding, or by an additional 34, or more, TS1

Ordered Sets when using 128b/130b encoding, but must not exceed 1 ms, so as not to prematurely configure a smaller Link than possible.

- Otherwise, after a $T_{crosslink}$ timeout, 16-32 TS2 Ordered Sets with PAD Link numbers and PAD Lane numbers are sent. The Upstream Lanes become Downstream Lanes and the next state is Configuration.Linkwidth.Start as Downstream Lanes.
 - Note: This optional behavior is required for crosslink behavior where two Ports may start off with Upstream Ports, and one will eventually take the lead as a Downstream Port.

- The next state is Detect after a 24 ms timeout.

4.2.6.3.2. Configuration.Linkwidth.Accept

4.2.6.3.2.1. Downstream Lanes

- If a configured Link can be formed with at least one group of Lanes that received two consecutive TS1 Ordered Sets with the same received Link number (non-PAD and matching one that was transmitted by the Downstream Lanes), TS1 Ordered Sets are transmitted with the same Link number and unique non-PAD Lane numbers are assigned to all these same Lanes. The next state is Configuration.Lanenum.Wait.
 - The assigned non-PAD Lane numbers must range from 0 to n-1, be assigned sequentially to the same grouping of Lanes that are receiving the same Link number, and Downstream Lanes which are not receiving TS1 Ordered Sets must not disrupt the initial sequential numbering of the widest possible Link. Any left over Lanes must transmit TS1 Ordered Sets with the Link and Lane number set to PAD.
 - It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set or lost 128b/130b Block Alignment on a subset of the received Lanes delay the evaluation listed above by an additional two, or more, TS1 Ordered Sets when using 8b/10b encoding, or by an additional 34, or more, TS1 Ordered Sets when using 128b/130b encoding, but must not exceed 1 ms, so as not to prematurely configure a smaller Link than possible.
- The next state is Detect after a 2 ms timeout or if no Link can be configured or if all Lanes receive two consecutive TS1 Ordered Sets with Link and Lane numbers set to PAD.

4.2.6.3.2.2. Upstream Lanes

- If a configured Link can be formed using Lanes that transmitted a non-PAD Link number which are receiving two consecutive TS1 Ordered Sets with the same non-PAD Link number and any non-PAD Lane number, TS1 Ordered Sets are transmitted with the same non-PAD Link number and Lane numbers that, if possible, match the received Lane numbers or are different, if necessary, (i.e., Lane reversed). The next state is Configuration.Lanenum.Wait.
 - The newly assigned Lane numbers must range from 0 to m-1, be assigned sequentially only to some continuous grouping of Lanes that are receiving non-PAD Lane numbers (i.e., Lanes which are not receiving any TS1 Ordered Sets always disrupt a continuous grouping and must not be included in this grouping), must include either Lane 0 or Lane

n-1 (largest received Lane number), and m-1 must be equal to or smaller than the largest received Lane number (n-1). Remaining Lanes must transmit TS1 Ordered Sets with Link and Lane numbers set to PAD.

- It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set or lost 128b/130b Block Alignment on a subset of the received Lanes delay the evaluation listed above by an additional two, or more, TS1 Ordered Sets when using 8b/10b encoding, or by an additional 34, or more, TS1 Ordered Sets when using 128b/130b encoding, but must not exceed 1 ms, so as not to prematurely configure a smaller Link than possible.

- The next state is Detect after a 2 ms timeout or if no Link can be configured or if all Lanes receive two consecutive TS1 Ordered Sets with Link and Lane numbers set to PAD.

IMPLEMENTATION NOTE

Example Cases

Notable examples related to the configuration of Downstream Lanes:

1. A x8 Downstream Port, which can be divided into two x4 Links, sends two different Link numbers on to two x4 Upstream Ports. The Upstream Ports respond simultaneously by picking the two Link numbers. The Downstream Port will have to choose one of these sets of Link numbers to configure as a Link, and leave the other for a secondary LTSSM to configure (which will ultimately happen in Configuration.Complete).
2. A x16 Downstream Port, which can be divided into two x8 Links, is hooked up to a x12 Upstream Port that can be configured as a x12 Link or a x8 and a x4 Link. During Configuration.Linkwidth.Start the Upstream Port returned the same Link number on all 12 Lanes. The Downstream Port would then return the same received Link number and assign Lane numbers on the eight Lanes that can form a x8 Link with the remaining four Lanes transmitting a Lane number and a Link number set to PAD.
3. A x8 Downstream Port where only seven Lanes are receiving TS1 Ordered Sets with the same received Link number (non-PAD and matching one that was transmitted by the Downstream Lanes) and an eighth Lane, which is in the middle or adjacent to those same Lanes, is not receiving a TS1 Ordered Set. In this case, the eighth Lane is treated the same as the other seven Lanes and Lane numbering for a x8 Lane should occur as described above.

Notable examples related to the configuration of Upstream Lanes:

A x8 Upstream Port is presented with Lane numbers that are backward from the preferred numbering. If the optional behavior of Lane reversal is supported by the Upstream Port, the Upstream Port transmits the same Lane numbers back to the Downstream Port. Otherwise the opposite Lane numbers are transmitted back to the Downstream Port, and it will be up to the Downstream Port to optionally fix the Lane ordering or exit Configuration.

> Optional Lane reversal behavior is required to configure a Link where the Lane numbers are reversed and the Downstream Port does not support Lane reversal. Specifically, the Upstream Port Lane reversal will accommodate the scenario where the default Upstream sequential Lane numbering (0 to n-1) is receiving a reversed Downstream sequential Lane number (n-1 to 0).

A x8 Upstream Port is not receiving TS1 Ordered Sets on the Upstream Port Lane 0:

a. In the case where the Upstream Port can only support a x8 or x1 Link and the Upstream Port can support Lane reversal. The Upstream Port will assign a Lane 0 to only the received Lane 7 (received Lane number n-1) and the remaining seven Lanes must transmit TS1 Ordered Sets with Link and Lane numbers set to PAD.
b. In the case where the Upstream Port can only support a x8 or x1 Link and the Upstream Port cannot support Lane reversal. No Link can be formed and the Upstream Port will eventually timeout after 2 ms and exit to Detect.

An optional x8 Upstream crosslink Port, which can be divided into two x4 Links, is attached to two x4 Downstream Ports that present the same Link number, and each x4 Downstream Port presents Lane numbers simultaneously that were each numbered 0 to 3. The Upstream Port will have to choose one of these sets of Lane numbers to configure as a Link, and leave the other for a second pass through Configuration.

4.2.6.3.3. Configuration.Lanenum.Accept

4.2.6.3.3.1. Downstream Lanes

- If two consecutive TS1 Ordered Sets are received with non-PAD Link and non-PAD Lane numbers that match all the non-PAD Link and non-PAD Lane numbers (or reversed Lane numbers if Lane reversal is optionally supported) that are being transmitted in Downstream Lane TS1 Ordered Sets, the next state is Configuration.Complete.
 - The Link Bandwidth Management Status and Link Autonomous Bandwidth Status bits of the Link Status register must be updated as follows on a Link bandwidth change if the current transition to Configuration state was from the Recovery state:
 (a) If the bandwidth change was initiated by the Downstream Port due to reliability issues, the Link Bandwidth Management Status bit is set to 1b.
 (b) Else if the bandwidth change was not initiated by the Downstream Port and the Autonomous Change bit (Symbol 4 bit 6) in two consecutive received TS1 Ordered Sets is 0b, the Link Bandwidth Management Status bit is set to 1b.
 (c) Else the Link Autonomous Bandwidth Status bit is set to 1b.
 - The condition of Reversed Lane numbers is defined strictly as the Downstream Lane 0 receiving a TS1 Ordered Set with a Lane number equal to n-1 and the Downstream Lane n-1 receiving a TS1 Ordered Set with a Lane number equal to 0.
 - It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set or lost 128b/130b Block Alignment on a subset of the received Lanes delay the evaluation listed above by an additional two, or more, TS1 Ordered Sets when using 8b/10b encoding, or by an additional 34, or more, TS1 Ordered Sets when using 128b/130b encoding, but must not exceed 1 ms, so as not to prematurely configure a smaller Link than possible.
- If a configured Link can be formed with any subset of the Lanes that receive two consecutive TS1 Ordered Sets with the same transmitted non-PAD Link numbers and any non-PAD Lane numbers, TS1 Ordered Sets are transmitted with the same non-PAD Link numbers and new Lane numbers assigned and the next state is Configuration.Lanenum.Wait.
 - The newly assigned transmitted Lane numbers must range from 0 to m-1, be assigned sequentially only to some continuous grouping of the Lanes that are receiving non-PAD Lane numbers (i.e., Lanes which are not receiving any TS1 Ordered Sets always disrupt a continuous grouping and must not be included in this grouping), must include either Lane 0 or Lane n-1 (largest received Lane number), and m-1 must be equal to or smaller than the largest received Lane number (n-1). Any left over Lanes must transmit TS1 Ordered Sets with the Link and Lane number set to PAD.
 - It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set or lost 128b/130b Block Alignment on a subset of the received Lanes delay the evaluation listed above by an additional two, or more, TS1 Ordered Sets when using 8b/10b encoding, or by an additional 34, or more, TS1 Ordered Sets when using

128b/130b encoding, but must not exceed 1 ms, so as not to prematurely configure a smaller Link than possible.

- The next state is Detect if no Link can be configured or if all Lanes receive two consecutive TS1 Ordered Sets with Link and Lane numbers set to PAD.

4.2.6.3.3.2. Upstream Lanes

- If two consecutive TS2 Ordered Sets are received with non-PAD Link and non-PAD Lane numbers that match all non-PAD Link and non-PAD Lane numbers that are being transmitted in Upstream Lane TS1 Ordered Sets, the next state is Configuration.Complete.
- If a configured Link can be formed with any subset of the Lanes that receive two consecutive TS1 Ordered Sets with the same transmitted non-PAD Link numbers and any non-PAD Lane numbers, TS1 Ordered Sets are transmitted with the same non-PAD Link numbers and new Lane numbers assigned and the next state is Configuration.Lanenum.Wait.
 - The newly assigned transmitted Lane numbers must range from 0 to m-1, be assigned sequentially only to some continuous grouping of Lanes that are receiving non-PAD Lane numbers (i.e., Lanes which are not receiving any TS1 Ordered Sets always disrupt a continuous grouping and must not be included in this grouping), must include either Lane 0 or Lane n-1 (largest received Lane number), and m-1 must be equal to or smaller than the largest received Lane number (n-1). Any left over Lanes must transmit TS1 Ordered Sets with the Link and Lane number set to PAD.
 - It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set or lost 128b/130b Block Alignment on a subset of the received Lanes delay the evaluation listed above by an additional two, or more, TS1 Ordered Sets when using 8b/10b encoding, or by an additional 34, or more, TS1 Ordered Sets when using 128b/130b encoding, but must not exceed 1 ms, so as not to pre-maturely configure a smaller Link than possible.
- The next state is Detect if no Link can be configured or if all Lanes receive two consecutive TS1 Ordered Sets with Link and Lane numbers set to PAD.

4.2.6.3.4. Configuration.Lanenum.Wait

4.2.6.3.4.1. Downstream Lanes

- The next state is Configuration.Lanenum.Accept if any of the Lanes that detected a Receiver during Detect receive two consecutive TS1 Ordered Sets which have a Lane number different from when the Lane first entered Configuration.Lanenum.Wait, and not all the Lanes' Link numbers are set to PAD or two consecutive TS1 Ordered Sets have been received on all Lanes, with Link and Lane numbers that match what is being transmitted on all Lanes.

 The Upstream Lanes are permitted delay up to 1 ms before transitioning to Configuration.Lanenum.Accept.

 The reason for delaying up to 1 ms before transitioning is to prevent received errors or skew between Lanes affecting the final configured Link width.

 The condition of requiring reception of any Lane number different from when the Lane(s) first entered Configuration.Lanenum.Wait is necessary in order to allow the two Ports to settle on an agreed upon Link width. The exact meaning of the statement "any of the Lanes receive two consecutive TS1 Ordered Sets, which have a Lane number different from when the Lane first entered Configuration.Lanenum.Wait" requires that a Lane number must have changed from when the Lanes most recently entered Configuration.Lanenum.Wait before a transition to Configuration.Lanenum.Accept can occur.

- The next state is Detect after a 2 ms timeout or if all Lanes receive two consecutive TS1 Ordered Sets with Link and Lane numbers set to PAD.

4.2.6.3.4.2. Upstream Lanes

- The next state is Configuration.Lanenum.Accept

 (a) If any of the Lanes receive two consecutive TS1 Ordered Sets that have a Lane number different from when the Lane first entered Configuration.Lanenum.Wait, and not all the Lanes' Link numbers are set to PAD

 or

 (b) If any Lane receives two consecutive TS2 Ordered Sets

- The next state is Detect after a 2 ms timeout or if all Lanes receive two consecutive TS1 Ordered Sets with Link and Lane numbers set to PAD.

4.2.6.3.5. Configuration.Complete

A device is allowed to change the supported data rates and upconfigure capability that it advertises when it enters this substate, but it must not change those values while in this substate.

4.2.6.3.5.1. Downstream Lanes

- TS2 Ordered Sets are transmitted using Link and Lane numbers that match the received TS1 Ordered Set Link and Lane numbers.

- The Upconfigure Capability bit of the TS2 Ordered Sets is permitted to be set to 1b to indicate that the Port is capable of supporting a x1 Link on the currently assigned Lane 0 and up-configuring the Link while LinkUp = 1b. Advertising this capability is optional.

- N_FTS must be noted for use in L0s when leaving this state.
- When using 8b/10b encoding, Lane-to-Lane de-skew must be completed when leaving this state.
- Scrambling is disabled if all configured Lanes have the Disable Scrambling bit asserted in two consecutively received TS2 Ordered Sets.
 - The Port that is sending the Disable Scrambling bit on all of the configured Lanes must also disable scrambling. Scrambling can only be disabled when using 8b/10b encoding.
- The next state is Configuration.Idle immediately after all Lanes that are transmitting TS2 Ordered Sets receive eight consecutive TS2 Ordered Sets with matching Lane and Link numbers (non-PAD) and identical data rate identifiers (including identical Link Upconfigure Capability (Symbol 4 bit 6)), and 16 TS2 Ordered Sets are sent after receiving one TS2 Ordered Set.
 - If the device supports greater than 2.5 GT/s data rate, it must record the data rate identifier received on any configured Lane of the Link. This will override any previously recorded value. A variable to track speed change in recovery state, changed_speed_recovery, is reset to 0b.
 - If the device sends TS2 Ordered Sets with the Link Upconfigure Capability (Symbol 4 bit 6) set to 1b, and receives eight consecutive TS2 Ordered Sets with the Link Upconfigure Capability bit set to 1b, the variable upconfigure_capable is set to 1b, else it is reset to 0b.
 - All remaining Lanes that are not part of the configured Link are no longer associated with the LTSSM in progress and must:

i. Be associated with a new LTSSM if this optional feature is supported.

or

ii. All Lanes that cannot be associated with an optional new LTSSM must transition to Electrical Idle.[52] Those Lanes that formed a Link up to the L0 state, and LinkUp has been 1b since then, but are not a part of the currently configured Link, must be associated with the same LTSSM if the LTSSM advertises Link width upconfigure capability. It is recommended that the Receiver terminations of these Lanes be left on. If they are not left on, they must be turned on when the LTSSM enters the Recovery.RcvrCfg substate until it reaches the Configuration.Complete substate if upconfigure_capable is set to 1b to allow for potential Link width upconfiguration. Any Lane that was not part of the LTSSM during the initial Link training through L0 cannot become a part of the LTSSM as part of the Link width upconfiguration process.

- In the case of an optional crosslink, the Receiver terminations are required to meet $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}POS}$ and $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}NEG}$ (see Table 4-24).

[52] The common mode being driven does not need to meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX\text{-}CM\text{-}DC\text{-}ACTIVE\text{-}IDLE\text{-}DELTA}$) specification (see Table 4-18).

- These Lanes must be re-associated with the LTSSM immediately after the LTSSM in progress transitions back to Detect.
- An EIOS does not need to be sent before transitioning to Electrical Idle, and the transition to Electrical Idle does not need to occur on a Symbol or Ordered Set boundary.

- After a 2 ms timeout:
 - The next state is Detect if the current data rate is 2.5 GT/s or 5.0 GT/s.
 - The next state is Configuration.Idle if the idle_to_rlock_transitioned variable is less than FFh and the current data rate is 8.0 GT/s.

i. The changed_speed_recovery variable is reset to 0b.

ii. Lanes that are not part of the configured Link are no longer associated with the LTSSM in progress and must meet requirement (i) or (ii) specified above for the non-timeout transition to Configuration.Idle.

iii. The upconfigure_capable variable is permitted, but not required, to be updated if at least one Lane received eight consecutive TS2 Ordered Sets with matching Lane and Link numbers (non-PAD). If updated, the upconfigure_capable variable is set to 1b when the transmitted and received Link Upconfigure Capability bits are 1b, else it is reset to 0b.

- Else the next state is Detect.

4.2.6.3.5.2. Upstream Lanes

- TS2 Ordered Sets are transmitted using Link and Lane numbers that match the received TS2 Link and Lane numbers.
 - The Upconfigure Capability bit of the TS2 Ordered Sets is permitted to be set to 1b to indicate that the Port is capable of supporting a x1 Link on the currently assigned Lane 0 and up-configuring the Link while LinkUp = 1b. Advertising this capability is optional.
- N_FTS must be noted for use in L0s when leaving this state.
- When using 8b/10b encoding, Lane-to-Lane de-skew must be completed when leaving this state.
- Scrambling is disabled if all configured Lanes have the Disable Scrambling bit asserted in two consecutively received TS2 Ordered Sets.
 - The Port that is sending the Disable Scrambling bit on all of the configured Lanes must also disable scrambling. Scrambling can only be disabled when using 8b/10b encoding.

- The next state is Configuration.Idle immediately after all Lanes that are transmitting TS2 Ordered Sets receive eight consecutive TS2 Ordered Sets with matching Lane and Link numbers (non-PAD) and identical data rate identifiers (including identical Link Upconfigure Capability (Symbol 4 bit 6)), and 16 consecutive TS2 Ordered Sets are sent after receiving one TS2 Ordered Set.
 - If the device supports greater than 2.5 GT/s data rate, it must record the data rate identifier received on any configured Lane of the Link. This will override any previously recorded value. A variable to track speed change in recovery state, changed_speed_recovery, is reset to 0b.
 - If the device sends TS2 Ordered Sets with the Link Upconfigure Capability (Symbol 4 bit 6) set to 1b, as well as receives eight consecutive TS2 Ordered Sets with the Link Upconfigure Capability bit set to 1b, the variable upconfigure_capable is set to 1b, else it is reset to 0b.
 - All remaining Lanes that are not part of the configured Link are no longer associated with the LTSSM in progress and must:

 i. Optionally be associated with a new crosslink LTSSM if this feature is supported.

 or

 ii. All remaining Lanes that are not associated with a new crosslink LTSSM must transition to Electrical Idle,[53] and Receiver terminations are required to meet $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}POS}$ and $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}NEG}$ (see Table 4-24). Those Lanes that formed a Link up to the L0 state, and LinkUp has been 1b since then, but are not a part of the currently configured Link, must be associated with the same LTSSM if the LTSSM advertises Link width upconfigure capability. It is recommended that the Receiver terminations of these Lanes be left on. If they are not left on, they must be turned on when the LTSSM enters the Recovery.RcvrCfg substate until it reaches the Configuration.Complete substate if upconfigure_capable is set to 1b to allow for potential Link width upconfiguration. Any Lane that was not part of the LTSSM during the initial Link training through L0 cannot become a part of the LTSSM as part of the Link width upconfiguration process.
 - These Lanes must be re-associated with the LTSSM immediately after the LTSSM in progress transitions back to Detect.
 - EIOS does not need to be sent before transitioning to Electrical Idle, and the transition to Electrical Idle does not need to occur on a Symbol or Ordered Set boundary.
- After a 2 ms timeout:
 - The next state is Detect if the current data rate is 2.5 GT/s or 5.0 GT/s.

[53] The common mode being driven does not need to meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX\text{-}CM\text{-}DC\text{-}ACTIVE\text{-}IDLE\text{-}DELTA}$) specification (see Table 4-18).

The next state is Configuration.Idle if the idle_to_rlock_transitioned variable is less than FFh and the current data rate is 8.0 GT/s.

i. The changed_speed_recovery variable is reset to 0b.

ii. Lanes that are not part of the configured Link are no longer associated with the LTSSM in progress and must meet requirement (i) or (ii) specified above for the non-timeout transition to Configuration.Idle.

iii. The upconfigure_capable variable is permitted, but not required, to be updated if at least one Lane received eight consecutive TS2 Ordered Sets with matching Lane and Link numbers (non-PAD). If updated, the upconfigure_capable variable is set to 1b when the transmitted and received Link Upconfigure Capability bits are 1b, else it is reset to 0b.

Else the next state is Detect.

4.2.6.3.6. Configuration.Idle

- When using 8b/10b encoding, the Transmitter sends Idle data Symbols on all configured Lanes. When using 128b/130b encoding, the Transmitter sends one SDS Ordered Set on all configured Lanes to start a Data Stream and then sends Idle data Symbols on all configured Lanes. The first Idle data Symbol transmitted on Lane 0 is the first Symbol of the Data Stream.
- Receiver waits for Idle data.
- LinkUp = 1b
- When using 8b/10b encoding, the next state is L0 if eight consecutive Symbol Times of Idle data are received on all configured Lanes and 16 Idle data Symbols are sent after receiving one Idle data Symbol.
 - If software has written a 1b to the Retrain Link bit in the Link Control register since the last transition to L0 from Recovery or Configuration, the Downstream Port must set the Link Bandwidth Management Status bit of the Link Status register to 1b.
- When using 128b/130b encoding, next state is L0 if eight consecutive Symbol Times of Idle data are received on all configured Lanes, 16 Idle data Symbols are sent after receiving one Idle data Symbol, and this state was not entered by a timeout from Configuration.Complete.
 - The Idle data Symbols must be received in Data Blocks.
 - Lane-to-Lane de-skew must be completed before Data Stream processing starts.
 - If software has written a 1b to the Retrain Link bit in the Link Control register since the last transition to L0 from Recovery or Configuration, the Downstream Port must set the Link Bandwidth Management Status bit of the Link Status register to 1b.
 - The idle_to_rlock_transitioned variable is reset to 00h on transition to L0.

- Otherwise, after a minimum 2 ms timeout:
 - If the idle_to_rlock_transitioned variable is less than FFh, the next state is Recovery.RcvrLock.
 - On transition to Recovery.RcvrLock:
 - If the data rate is 8.0 GT/s, the idle_to_rlock_transitioned variable is incremented by 1.
 - If the data rate is 2.5 GT/s or 5.0 GT/s, the idle_to_rlock_transitioned variable is set to FFh.
 - Else the next state is Detect.

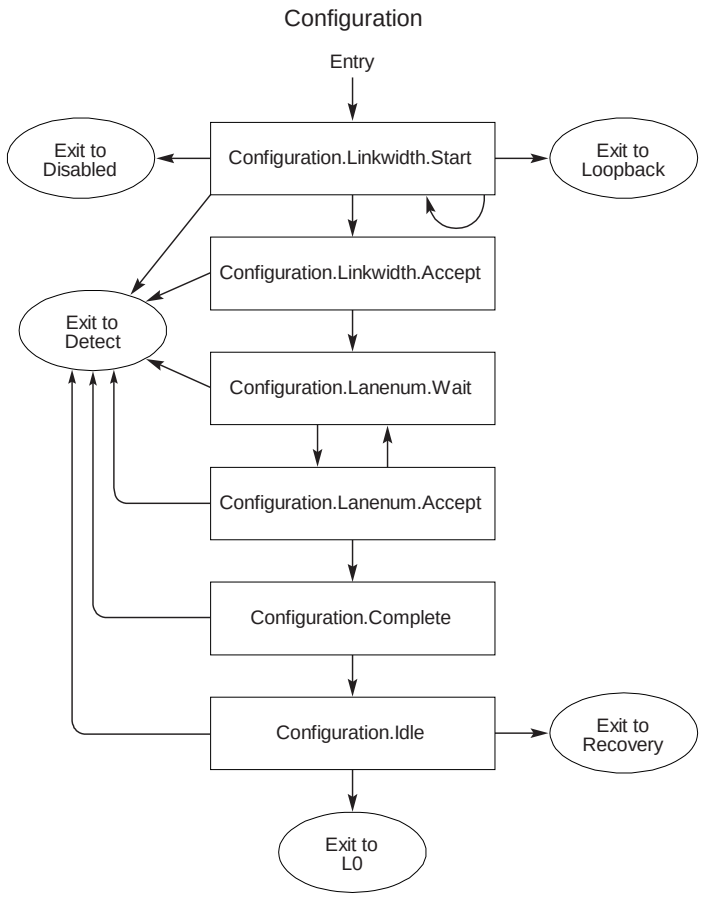


Figure 4-25: Configuration Substate Machine

4.2.6.4. Recovery

The Recovery substate machine is shown in Figure 4-26

4.2.6.4.1. Recovery.RcvrLock

If the Link is operating at 8.0 GT/s data rate, a Receiver must consider any TS1 or TS2 Ordered Set to be received only after it obtains Block Alignment in that Lane. If entry to this substate is from L1 or Recovery.Speed or L0s, the Block Alignment must be obtained after exiting Electrical Idle condition. If entry to this substate is from L0, the Block Alignment must be obtained after the end of the last Data Stream.

- If the data rate of operation is 8.0 GT/s:
 - If the start_equalization_w_preset variable is set to 1b:
 - An Upstream Port must use the preset values it registered from the received eight consecutive TS2 Ordered Sets in Recovery.RcvrCfg in its Transmitter setting as soon as it starts transmitting in 8.0 GT/s data rate and ensure that it meets the preset definition in Section 4.3. It may optionally use the Receiver preset setting suggested by the Downstream Port in those TS2 Ordered Sets. (Note: These are EQ TS2 Ordered Sets.)
 - A Downstream Port must use the preset values defined in the Lane Equalization Control register as soon as it starts transmitting in 8.0 GT/s data rate and ensure that it meets the preset definition in Section 4.3. It may optionally use the preset values defined for its Receivers.
 - Next state is Recovery.Equalization.
 - Else:
 - The Transmitter must use the coefficient settings agreed upon at the conclusion of the last equalization procedure
 - If this substate was entered from Recovery.Equalization, in the transmitted TS1 Ordered Sets, a Downstream Port must set the Pre-cursor, Cursor, and Post-cursor Coefficient fields to the current Transmitter settings, and if the last accepted request in Phase 2 of Recovery.Equalization was a preset request, it must set the Transmitter Preset field to the accepted preset of that request.
 - It is recommended that in this substate, in the transmitted TS1 Ordered Sets, all Ports set the Pre-cursor, Cursor, and Post-cursor Coefficient fields to the current Transmitter settings, and set the Transmitter Preset field to the most recent preset that the Transmitter settings were set to.
 - An Upstream Port that receives eight consecutive TS1 Ordered Sets on all configured Lanes with the following characteristics must transition to Recovery.Equalization
 - Link and Lane numbers in the received TS1 Ordered Sets match with the Link and Lane numbers in the transmitted TS1 Ordered Sets on each Lane

Comment [VS123]: Errata A15

- speed_change bit is equal to 0b
- EC bits not equal to 00b



IMPLEMENTATION NOTE

Redoing Equalization

A Downstream Port may use this provision to redo some parts of the Transmitter Equalization process using software help or some other implementation specific means while ensuring no transactions are in flight on the Link to avoid any timeouts.

- Next state for a Downstream Port is Recovery.Equalization if directed and Recovery.RcvrLock was not entered from Configuration.Idle or Recovery.Idle. The Port must ensure that no more than 2 TS1 Ordered Sets with EC=00b are transmitted due to being in Recovery.RcvrLock before starting to transmit the TS1 Ordered Sets required by Recovery.Equalization.

- Transmitter sends TS1 Ordered Sets on all configured Lanes using the same Link and Lane numbers that were set after leaving Configuration. The speed_change bit (bit 7 of the Data Rate Identifier Symbol in TS1 Ordered Set) must be set to 1b if the directed_speed_change variable is set to 1b. The directed_speed_change variable is set to 1b if any configured Lane receives eight consecutive TS1 Ordered Sets with the speed_change bit set to 1b. Only those data rates greater than 2.5 GT/s should be advertised that can be supported reliably. The N_FTS value in the TS1 Ordered Set transmitted reflects the number at the current speed of operation. A device is allowed to change the supported data rates that it advertises when it enters this substate. A Downstream Port that intends to redo equalization with a data rate change to 8.0 GT/s must send EQ TS1 Ordered Sets with the speed_change bit set to 1b and advertising the 8.0 GT/s Data Rate Identifier. An Upstream Port is expected to advertise 8.0 GT/s data rate support in the TS2 Ordered Sets it transmits in Recovery.RcvrCfg, and optionally in the TS1 Ordered Sets it transmits in this substate, if the eight consecutive Ordered Sets it receives are EQ TS1 or EQ TS2 Ordered Sets with the speed_change bit set to 1b and 8.0 GT/s data rate support is advertised by the Downstream Port, unless the Upstream Port has determined that the reliability problems at the 8.0 GT/s data rate were unrelated to equalization. Under other conditions, a device must not change the supported data rate values either in this substate or while in the Recovery.RcvrCfg or Recovery.Equalization substates. The successful_speed_negotiation variable is reset to 0b upon entry to this substate.



IMPLEMENTATION NOTE

Handling a Request to Advertise 8.0 GT/s Data Rate Identifier

If an Upstream Port that is not advertising 8.0 GT/s Data Rate Identifiers receives EQ TSs with 8.0 GT/s Data Rate Identifiers and with the speed_change bit set in Recovery.RcvrLock, that indicates that the Downstream Port is attempting to switch the Link Speed to 8 GT/s in order to perform the 8.0 GT/s Link Equalization Procedure. If for some reason the Upstream Port is unable or unwilling to switch to advertising 8.0 GT/s Data Rate Identifiers in the TS2 Ordered Sets it transmits once it transitions to Recovery.RcvrCfg, the 8.0 GT/s Link Equalization Procedure will not be performed in the current tenure in Recovery. This may cause the Downstream Port to permanently abandon its attempt to change the link speed to 8.0 GT/s and perform the 8.0 GT/s Link Equalization Procedure, resulting in an operational link speed of less than 8.0 GT/s until after the link transitions through Detect and is re-trained.

It is recommended that if an Upstream Port is for some temporary reason unable or unwilling to switch to advertising 8.0 GT/s Data Rate Identifiers in the condition described above, and does not intend to prohibit the Link from operating at 8.0 GT/s, that it perform one of the following two actions below as soon as is reasonable for it to do so:

- If the Upstream Port supports the Quiesce Guarantee mechanism for performing the Link Equalization Procedure, enter Recovery and advertise 8.0 GT/s Data Rate Identifiers with the speed_change bit set to 1b in the TSs that it sends. If Recovery.Equalization is not entered after changing speed to 8.0 GT/s and before entering Recovery.RcvrCfg at 8.0 GT/s (the Downstream Port did not direct an entry to Recovery.Equalization), it should set the Request Equalization and Quiesce Guarantee bits to 1b in the TS2 Ordered Sets sent at 8.0 GT/s in Recovery.RcvrCfg in order to request the Downstream Port to initiate the Link Equalization Procedure.
- Enter Recovery and advertise 8.0 GT/s Data Rate Identifiers with the speed_change bit cleared to 0b. The Downstream Port may then later initiate a speed change to 8.0 GT/s and perform the Link Equalization Procedure, though there is no guarantee that it will do so.

An Upstream Port must set the Selectable De-emphasis bit (bit 6 of Symbol 4) of the TS1 Ordered Sets it transmits to match the desired de-emphasis level at 5.0 GT/s. The mechanism an Upstream Port may adopt to request a de-emphasis level if it chooses to do so is implementation specific. It must also be noted that since the Upstream Port's request may not reach the Downstream Port due to bit errors in the TS1 Ordered Sets, the Upstream Port may attempt to re-request the desired de-emphasis level in subsequent entries to Recovery state when speed change is requested. If the Downstream Port intends to use the Upstream Port's de-emphasis information in Recovery.RcvrCfg, then it must record the value of the Selectable De-emphasis bit received in this state.

The Transmit Margin field of the Link Control 2 register is sampled on entry to this substate and becomes effective on the transmit package pins within 192 ns of entry to this substate and remains effective until a new value is sampled on a subsequent entry to this substate from L0, L0s, or L1.

- After activating any inactive Lane, the Transmitter must wait for its TX common mode to settle before exiting Electrical Idle and transmitting the TS1 Ordered Sets with the following exceptions.
- When exiting from the L1.2 L1 PM Substate, common mode is permitted to be established passively during L1.0, and actively during Recovery. In order to ensure common mode has been established in Recovery.RcvrLock, the Downstream Port must maintain a timer, and the Downstream Port must not send TS2 training sequences until a minimum of $T_{COMMONMODE}$ has elapsed since the Downstream Port has started both transmitting and receiving TS1 training sequences. See Section 5.5.3.3.
- Implementations must note that the voltage levels may change after an early bit lock and Symbol or Block alignment since the new Transmit Margin field becomes effective within 192 ns after the other side enter Recovery.RcvrLock. The Receiver needs to reacquire bit lock and Symbol or Block alignment under those conditions.
 a. Note: The directed_speed_change variable is set to 1b in L0 or L1 state for the side that is initiating a speed change. For the side that is not initiating a speed change, this bit is set to 1b in this substate if the received TS Ordered Sets have the speed change bit set to 1b. This bit is reset to 0b in the Recovery.Speed substate.
 b. A device must accept all good TLPs and DLLPs it receives after entering this substate from L0 prior to receiving the first TS Ordered Set. If operating with 128b/130b encoding, any received TLPs and DLLPs are subject to the framing rules for 128b/130b encoding in Section 4.2.2.3.

- Next state is Recovery.RcvrCfg if eight consecutive TS1 or TS2 Ordered Sets are received on all configured Lanes with the same Link and Lane numbers that match what is being transmitted on those same Lanes and the speed_change bit is equal to the directed_speed_change variable and the EC field is 00b in all the consecutive TS1 Ordered Sets if the current data rate is 8.0 GT/s.
 - If the Extended Synch bit is set, the Transmitter must send a minimum of 1024 consecutive TS1 Ordered Sets before transitioning to Recovery.RcvrCfg.
 - If this substate was entered from Recovery.Equalization, the Upstream Port must evaluate the equalization coefficients or preset received by all Lanes that receive eight TS1 Ordered Sets and note whether they are different from the final set of coefficients or preset that was accepted in Phase 2 of the equalization process. Note: Mismatches are reported in Recovery.RcvrCfg by setting the Request Equalization bit of TS2 Ordered Sets.
- Otherwise, after a 24 ms timeout:
 - Next state is Recovery.RcvrCfg if the following two conditions are true:
 - Eight consecutive TS1 or TS2 Ordered Sets are received on any configured Lane with the same Link and Lane numbers that match what is being transmitted on the same Lane and the speed_change bit equal to 1b.
 - Either the current data rate of operation is greater than 2.5 GT/s; or 5.0 GT/s or greater data rate identifiers are set in both the transmitted TS1 and the (eight consecutive) received TS1 or TS2 Ordered Sets.

- Else the next state is Recovery.Speed if the speed of operation has not changed to a mutually negotiated data rate since entering Recovery from L0 or L1 (i.e., changed_speed_recovery = 0b) and the current speed of operation is greater than 2.5 GT/s. The new data rate to operate after leaving Recovery.Speed will be at 2.5 GT/s.
 Note: This indicates that the Link was unable to operate at the current data rate (greater than 2.5 GT/s) and the Link will operate at the 2.5 GT/s data rate.
- Else the next state is Recovery.Speed if the operating speed has been changed to a mutually negotiated data rate since entering Recovery from L0 or L1 (changed_speed_recovery = 1b; i.e., the arc to this substate has been taken from Recovery.Speed). The new data rate to operate after leaving Recovery.Speed is reverted back to the speed it was when Recovery was entered from L0 or L1.

 Note: This indicates that the Link was unable to operate at the new negotiated data rate and will revert back to the old data rate with which it entered Recovery from L0 or L1.
- Else the next state is Configuration and the directed_speed_change variable is reset to 0b if any of the configured Lanes that are receiving a TS1 or TS2 Ordered Set have received at least one TS1 or TS2 Ordered Set with Link and Lane numbers that match what is being transmitted on those same Lanes and the operating speed has not changed to a mutually negotiated data rate (i.e., changed_speed_recovery = 0b) since entering Recovery and at least one of the following conditions is true:
 - The directed_speed_change variable is equal to 0b and the speed_change bit on the received TS1 or TS2 Ordered Set is equal to 0b.
 - The current data rate of operation is 2.5 GT/s and 2.5 GT/s data rate is the highest commonly advertised data rate among the transmitted TS1 Ordered Sets and the received TS1 or TS2 Ordered Set(s).
- Otherwise, the next state is Detect.

IMPLEMENTATION NOTE

Example Showing Speed Change Algorithm Between 2.5 GT/s and 5.0 GT/s

Suppose a Link connects two greater than 5.0 GT/s capable components, A and B. The Link comes up to the L0 state in 2.5 GT/s data rate. Component A decides to change the speed to greater than 5.0 GT/s, sets the directed_speed_change variable to 1b and enters Recovery.RcvrLock from L0. Component A sends TS1 Ordered Sets with the speed_change bit set to 1b and advertises all three data rates. Component B sees the first TS1 in L0 state and enters Recovery.RcvrLock state. Initially, component B sends TS1s with the speed_change set to 0b. Component B will start sending the speed_change indication in its TS1 after it receives eight consecutive TS1 Ordered Sets from component A and advertises all three data rates it can support. Component B will enter Recovery.RcvrCfg from where it will enter Recovery.Speed. Component A will wait for eight consecutive TS1/TS2 with speed_change bit set from component B before moving to Recovery.RcvrCfg and on to Recovery.Speed. Both component A and component B enter Recovery.Speed and record 8.0 GT/s as the maximum speed they can operate with. The directed_speed_change variable will be reset to 0b when in Recovery.Speed. When they enter Recovery.RcvrLock from Recovery.Speed, they will operate at 8.0 GT/s and send TS1s with speed_change set to 0b. If both sides work well at 8.0 GT/s, they will continue on to Recovery.RcvrCfg and enter L0 through Recovery.Idle at 8.0 GT/s. However, if component B fails to achieve Symbol lock, it will timeout in Recovery.RcvrLock and enters Recovery.Speed. Component A would have moved on to Recovery.RcvrCfg but would see the Electrical Idle after receiving TS1s at 8.0 GT/s after component B enters Recovery.Speed. This will cause component A to move to Recovery.Speed. After entering Recovery.Speed for the second time, both sides will revert back to the speed they operated with prior to entering the Recovery state (2.5 GT/s). Both sides will enter L0 from Recovery in 2.5 GT/s. Component A may initiate the directed_speed_change variable for a second time, requesting 8.0 GT/s data rate in its Data Rate Identifier, go through the same steps, fail to establish the 8.0 GT/s data rate and go back to L0 in 2.5 GT/s data rate. On the third attempt, however, component A may decide to only advertise 2.5 GT/s and 5.0 GT/s rates and successfully establish the Link at 5.0 GT/s data rate and enter L0 at that speed. However, if either side entered Detect, it should advertise 2.5 GT/s through 8.0 GT/s capability since there may have been a hot plug event.

4.2.6.4.2. Recovery.Equalization

Transmitter sends TS1 Ordered Sets on all configured Lanes using the same Link and Lane numbers that were set after leaving Configuration.

4.2.6.4.2.1 Downstream Lanes

Upon entry to this substate:

- Current phase is Phase 1
- The Equalization Phase 1 Successful, Equalization Phase 2 Successful, Equalization Phase 3 Successful, Link Equalization Request, and Equalization Complete bits of the Link Status 2 register and the Perform Equalization bit of the Link Control 3 register are all set to 0b
- The equalization_done_8GT_data_rate variable is set to 1b
- The start_equalization_w_preset variable is set to 0b

4.2.6.4.2.1.1. Phase 1 of Transmitter Equalization

- Transmitter sends TS1 Ordered Sets using the Transmitter settings specified by the Downstream Port Transmitter Preset fields of the Lane Equalization Control register. In the TS1 Ordered Sets, the EC field is set to 01b, the Transmitter Preset field of each Lane is set to the value of its corresponding Downstream Transmitter Preset, and the FS, LF, and Post-cursor Coefficient fields are set to values corresponding to the Transmitter Preset field.
- The Downstream Port is permitted to wait for up to 500 ns after entering Phase 1 before evaluating received information for TS1 Ordered Sets if it needs the time to stabilize its Receiver logic.
- Next phase is Phase 2 if all configured Lanes receive two consecutive TS1 Ordered Sets with EC=01b and the Downstream Port wants to execute Phase 2 and Phase 3.
 - The Receiver must complete its bit lock process and then recognize Ordered Sets within 2 ms after receiving the first bit of the first valid Ordered Set on its Receiver pin.
 - The Equalization Phase 1 Successful bit of the Link Status 2 register is set to 1b.
 - The LF and FS values received in the two consecutive TS1 Ordered Sets must be stored for use during Phase 3, if the Downstream Port wants to adjust the Upstream Port's Transmitter coefficients.

- Else, next state is Recovery.RcvrLock if all configured Lanes receive two consecutive TS1 Ordered Sets with EC=01b and the Downstream Port does not want to execute Phase 2 and Phase 3.
 - The Equalization Phase 1 Successful, Equalization Phase 2 Successful, Equalization Phase 3 Successful, and Equalization Complete bits of the Link Status 2 register are set to 1b.
 - Note: A transition to Recovery.RcvrLock might be used in the case where the Downstream Port determines that Phase 2 and Phase 3 are not needed based on the platform and channel characteristics.
- Else, next state is Recovery.Speed after a 24 ms timeout.
 - successful_speed_negotiation is set to 0b
 - The Equalization Complete bit of the Link Status 2 register is set to 1b.

4.2.6.4.2.1.2. Phase 2 of Transmitter Equalization

- Transmitter sends TS1 Ordered Sets with EC = 10b and the coefficient settings, set on each Lane independently, as follows:
 - If two consecutive TS1 Ordered Sets with EC=10b have been received since entering Phase 2, or two consecutive TS1 Ordered Sets with EC=10b and a preset or set of coefficients (as specified by the Use Preset bit) different than the last two consecutive TS1 Ordered Sets with EC=10b:
 - If the preset or coefficients requested in the most recent two consecutive TS1 Ordered Sets are legal and supported (see Section 4.2.3):
 - Change the transmitter settings to the requested preset or coefficients such that the new settings are effective at the Transmitter pins within 500 ns of when the end of the second TS1 Ordered Set requesting the new setting was received at the Receiver pin. The change of Transmitter settings must not cause any illegal voltage level or parameter at the Transmitter pin for more than 1 ns.
 - ~~Reflect the new settings i~~In the transmitted TS1 Ordered Sets, the Transmitter Preset field is set to the requested preset (for a preset request), the Pre-cursor, Cursor, and Post-cursor Coefficient fields are set to the Transmitter settings (for a preset or a coefficients request), and ~~clear~~ the Reject Coefficient Values bit is set to 0b.
 - Else (the requested preset or coefficients are illegal or unsupported): Do not change the Transmitter settings used, but reflect the requested preset or coefficient values in the transmitted TS1 Ordered Sets and set the Reject Coefficient Values bit to 1b.
 - Else: the preset and coefficients currently being used by the Transmitter.
- Next phase is Phase 3 if all configured Lanes receive two consecutive TS1 Ordered Sets with EC=11b.

Comment [VS124]: Errata A15

- The Equalization Phase 2 Successful bit of the Link Status 2 register is set to 1b.

- Else, next state is Recovery.Speed after a 32 ms timeout with a tolerance of -0 ms and +4 ms
 - successful_speed_negotiation is set to 0b.
 - The Equalization Complete bit of the Link Status 2 register is set to 1b.

4.2.6.4.2.1.3. Phase 3 of Transmitter Equalization

- Transmitter sends TS1 Ordered Sets with EC = 11b
- The Port must evaluate and arrive at the optimal settings independently on each Lane. To evaluate a new preset or coefficient setting that is legal, as per the rules in Section 4.2.3 and 4.3:
 - Request a new preset by setting the Transmitter Preset field to the desired value and set the Use Preset bit to 1b. Or, request a new set of coefficients by setting the Pre-cursor, Cursor, and Post-Cursor Coefficient fields to the desired values and set the Use Preset bit to 0b. Once a request is made, it must be continuously requested for at least 1 µs or until the evaluation of the request is completed, whichever is later.
 - Wait for the required time (500 ns plus the roundtrip delay including the logic delays through the Downstream Port) to ensure that, if accepted, the Upstream Port is transmitting using the requested settings. Obtain Block Alignment and then evaluate the incoming Ordered Sets. Note: The Downstream Port may simply ignore anything it receives during this waiting period as the incoming bit stream may be illegal during the transition to the requested settings. Hence the requirement to validate Block Alignment after this waiting period.
 - If two consecutive TS1 Ordered Sets are received with the Transmitter Preset field (for a preset request) or the Pre-cursor, Cursor, and Post-Cursor fields (for a coefficients request) identical to what was requested and the Reject Coefficient Values bit is set to 0b, then the requested setting was accepted and can be considered for evaluation. If two consecutive TS1 Ordered Sets are received with the Transmitter Preset field (for a preset request) or the Pre-cursor, Cursor, and Post-Cursor fields (for a coefficients request) identical to what was requested and the Reject Coefficient Values bit is set to 1b, then the requested setting was rejected. It is recommended that the Downstream Port request a different preset or set of coefficients. However, the Downstream Port is permitted to exit this phase and not make any additional requests. The Downstream Port is responsible for setting the Reset EIEOS Interval Count bit in the TS1 Ordered Sets it transmits according to its evaluation criteria and requirements. The Use Preset bit of the received TS1 Ordered Sets must not be used to determine whether a request is accepted or rejected.

Comment [VS125]: Errata A4

IMPLEMENTATION NOTE

Reset EIEOS and Coefficient/ Preset Requests

A Port may set Reset EIEOS Interval Count to 1b when it wants a longer PRBS pattern and subsequently clear it when it needs to obtain Block Alignment.

All TS1 Ordered Sets transmitted in this Phase are requests. The first request maybe a new preset or a new coefficient request or a request to maintain the current link partner transmitter settings by reflecting the settings received in the two consecutive TS1 Ordered Sets with EC=11b that cause the transition to Phase 3.

- The total amount of time spent per preset or coefficients request from transmission of the request to the completion of the evaluation must be less than 2 ms. Implementations that need a longer evaluation time at the final stage of optimization may continue requesting the same preset or coefficient setting beyond the 2 ms limit but must adhere to the 24 ms timeout in this Phase and must not take this exception more than two times. If the requester is unable to receive Ordered Sets within the timeout period, it may assume that the requested setting does not work in that Lane.
- All new preset or coefficient settings must be presented on all configured Lanes simultaneously. Any given Lane is permitted to continue to transmit the current preset or coefficients as its new value if it does not want to change the setting at that time.

- Next state is Recovery.RcvrLock if all configured Lanes are operating at their optimal settings.
 - The Equalization Phase 3 Successful and Equalization Complete bits of the Link Status 2 register are set to 1b
- Else, next state is Recovery.Speed after a timeout. of 24 ms with a tolerance of -0 ms and +2 ms
 - successful_speed_negotiation is set to 0b
 - The Equalization Complete bit of the Link Status 2 register is set to 1b.

4.2.6.4.2.2 Upstream Lanes

Upon entry to this substate:

- Current phase is Phase 0
- The Equalization Phase 1 Successful, Equalization Phase 2 Successful, Equalization Phase 3 Successful, Link Equalization Request, and Equalization Complete bits of the Link Status 2 register are all set to 0b
- The equalization_done_8GT_data_rate variable is set to 1b
- The start_equalization_w_preset variable is set to 0b.

4.2.6.4.2.2.1. Phase 0 of Transmitter Equalization

- Transmitter sends TS1 Ordered Sets using the Transmitter settings specified by the Transmitter Presets received in the EQ TS2 Ordered Sets during the most recent transition to 8.0 GT/s data rate from 2.5 GT/s or 5.0 GT/s data rate. In the TS1 Ordered Sets, the EC field is set to 00b, the Transmitter Preset field of each Lane is set to the value it received in the EQ TS2 Ordered Sets, and the Pre-cursor Coefficient, Cursor Coefficient, and Post-cursor Coefficient fields are set to values corresponding to the Transmitter Preset field.
 - Lanes that received a Reserved or unsupported Transmitter Preset in the EQ TS2 Ordered Sets: In the TS1 Ordered Sets, the Transmitter Preset field is set to the received Transmitter Preset, but the Coefficient fields are set to values corresponding to the implementation-specific Transmitter Preset chosen by the Lane.
 - Lanes that did not receive EQ TS2 Ordered Sets: In the TS1 Ordered Sets, the Transmitter Preset field is undefined, and the Coefficient fields are set to values corresponding to the implementation-specific Transmitter Preset chosen by the Lane.
- The Upstream Port is permitted to wait for up to 500 ns after entering Phase 0 before evaluating receiver information for TS1 Ordered Sets if it needs the time to stabilize its Receiver logic.
- Next phase is Phase 1 if all the configured Lanes receive two consecutive TS1 Ordered Sets with EC=01b.
 - The Receiver must complete its bit lock process and then recognize Ordered Sets within 2 ms after receiving the first bit of the first valid Ordered Set on its Receiver pin.
 - The LF and FS values received in the two consecutive TS1 Ordered Sets must be stored for use during Phase 2 if the Upstream Port wants to adjust the Downstream Port's Transmitter coefficients.
- Else, next state is Recovery.Speed after a 12 ms timeout.
 - successful_speed_negotiation is set to 0b
 - The Equalization Complete bit of the Link Status 2 register is set to 1b.

4.2.6.4.2.2.2. Phase 1 of Transmitter Equalization

- Transmitter sends TS1 Ordered Sets using the Transmitter settings determined in Phase 0. In the TS1 Ordered Sets, the EC field is set to 01b, and the FS, LF, and Post-cursor Coefficient fields of each Lane are set to values corresponding to the Lane's current Transmitter settings.
- Next phase is Phase 2 if all configured Lanes receive two consecutive TS1 Ordered Sets with EC=10b
 - The Equalization Phase 1 Successful bit of the Link Status 2 register is set to 1b
- Next state is Recovery.RcvrLock if all configured Lanes receive eight consecutive TS1 Ordered Sets with EC=00b
 - The Equalization Phase 1 Successful and Equalization Complete bits of the Link Status 2 register are set to 1b
- Else, next state is Recovery.Speed after a 12 ms timeout

- successful_speed_negotiation is set to 0b
- The Equalization Complete bit of the Link Status 2 register is set to 1b.

4.2.6.4.2.2.3. Phase 2 of Transmitter Equalization

- Transmitter sends TS1 Ordered Sets with EC = 10b
- The Port must evaluate and arrive at the optimal settings independently on each Lane. To evaluate a new preset or coefficient setting that is legal, as per the rules in Section 4.2.3 and 4.3:
 - Request a new preset by setting the Transmitter Preset field to the desired value and set the Use Preset bit to 1b. Or, request a new set of coefficients by setting the Pre-cursor, Cursor, and Post-cursor Coefficient fields to the desired values and set the Use Preset bit to 0b. Once a request is made, it must be continuously requested for at least 1 µs or until the evaluation of the request is completed, whichever is later.
 - Wait for the required time (500 ns plus the roundtrip delay including the logic delays through the Upstream Port) to ensure that, if accepted, the Downstream Port is transmitting using the requested settings. Obtain Block Alignment and then evaluate the incoming Ordered Sets. Note: The Upstream Port may simply ignore anything it receives during this waiting period as the incoming bit stream may be illegal during the transition to the requested settings. Hence the requirement to validate Block Alignment after this waiting period.
 - If two consecutive TS1 Ordered Sets are received with the Transmitter Preset field (for a preset request) or the Pre-cursor, Cursor, and Post-Cursor fields (for a coefficients request) identical to what was requested and the Reject Coefficient Values bit is set to 0b, then the requested setting was accepted and can be considered for evaluation. If two consecutive TS1 Ordered Sets are received with the Transmitter Preset field (for a preset request) or the Pre-Cursor, Cursor, and Post-Cursor fields (for a coefficients request) identical to what was requested and the Reject Coefficient Values bit is set to 1b, then the requested setting was rejected. It is recommended that the Upstream Port request a different preset or set of coefficients. However, the Upstream Port is permitted to exit this phase and not make any additional requests. The Upstream Port is responsible for setting the Reset EIEOS Interval Count bit in the TS1 Ordered Sets it transmits according to its evaluation criteria and requirements. The Use Preset bit of the received TS1 Ordered Sets must not be used to determine whether a request is accepted or rejected.

Comment [VS126]: Errata A4



IMPLEMENTATION NOTE

Reset EIEOS and Coefficient/Preset Requests

A Port may set Reset EIEOS Interval Count to 1b when it wants a longer PRBS pattern and subsequently clear it when it needs to obtain Block Alignment.

All TS1 Ordered Sets transmitted in this Phase are requests. The first request maybe a new preset or a new coefficient request or a request to maintain the current link partner transmitter settings by reflecting the settings received in the two consecutive TS1 Ordered Sets with EC=10b that cause the transition to Phase 2.

- The total amount of time spent per preset or coefficients request from transmission of the request to the completion of the evaluation must be less than 2 ms. Implementations that need a longer evaluation time at the final stage of optimization may continue requesting the same setting beyond the 2 ms limit but must adhere to the 24 ms timeout in this Phase and must not take this exception more than two times. If the requester is unable to receive Ordered Sets within the timeout period, it may assume that the requested setting does not work in that Lane.
- All new preset or coefficient settings must be presented on all configured Lanes simultaneously. Any given Lane is permitted to continue to transmit the current preset or coefficients as its new value if it does not want to change the setting at that time.

❑ Next phase is Phase 3 if all configured Lanes are operating in their optimal settings.

- The Equalization Phase 2 Successful bit of the Link Status 2 register is set to 1b

❑ Else, next state is Recovery.Speed after a 24 ms timeout with a tolerance of -0 ms and +2 ms

- successful_speed_negotiation is set to 0b
- The Equalization Complete bit of the Link Status 2 register is set to 1b.

4.2.6.4.2.2.4. Phase 3 of Transmitter Equalization

❑ Transmitter sends TS1 Ordered Sets with EC = 11b and the coefficient settings, set on each configured Lane independently, as follows:

- If two consecutive TS1 Ordered Sets with EC=11b have been received since entering Phase 3, or two consecutive TS1 Ordered Sets with EC=11b and a preset or set of coefficients (as specified by the Use Preset bit) different than the last two consecutive TS1 Ordered Sets with EC=11b:
 - If the preset or coefficients requested in the most recent two consecutive TS Ordered Sets are legal and supported (see Section 4.2.3 and 4.3):
 - Change the transmitter settings to the requested preset or coefficients such that the new settings are effective at the Transmitter pins within 500 ns of when the end of the second TS1 Ordered Set requesting the new setting was received at the Receiver pin. The change of Transmitter settings must not cause any illegal voltage level or parameter at the Transmitter pin for more than 1 ns.
 - ~~Reflect the new requested settings i~~In the transmitted TS1 Ordered Sets, the Transmitter Preset field is set to the requested preset (for a preset request), the Pre-cursor, Cursor, and Post-cursor Coefficient fields are set to the Transmitter settings (for a preset or a coefficients request), and ~~clear~~ the Reject Coefficient Values bit is set to 0b.
 - Else (the requested preset or coefficients are illegal or unsupported): Do not change the Transmitter settings used, but reflect the requested preset or coefficient values in the transmitted TS1 Ordered Sets and set the Reject Coefficient Values bit to 1b.

Comment [VS127]: Errata A15

- Else: the preset and coefficients currently being used by the Transmitter.
 - The preset initially transmitted on entry to Phase 3 can be the Transmitter Preset transmitted in Phase 0 or the preset currently being used by the Transmitter.

- Next state is Recovery.RcvrLock if all configured Lanes receive two consecutive TS1 Ordered Sets with EC=00b.
 - The Equalization Phase 3 Successful and Equalization Complete bits of the Link Status 2 register are set to 1b
- Else, next state is Recovery.Speed after a timeout of 32 ms with a tolerance of -0 ms and +4 ms
 - successful_speed_negotiation is set to 0b
 - The Equalization Complete bit of the Link Status 2 register is set to 1b.

4.2.6.4.3. Recovery.Speed

- The Transmitter enters Electrical Idle and stays there until the Receiver Lanes have entered Electrical Idle, and then additionally remains there for at least 800 ns on a successful speed negotiation (i.e., successful_speed_negotiation = 1b) or at least 6 µs on an unsuccessful speed negotiation (i.e., successful_speed_negotiation = 0b), but stays there no longer than an additional 1 ms. The frequency of operation is permitted to be changed to the new data rate only after the Receiver Lanes have entered Electrical Idle. If the negotiated data rate is 5.0 GT/s, and if operating in full swing mode, -6 dB de-emphasis level must be selected for operation if the select_deemphasis variable is 0b and -3.5 dB de-emphasis level must be selected for operation if the select_deemphasis variable is 1b. Note that if the link is already operating at the highest data rate supported by both Ports, Recovery.Speed is executed but the data rate is not changed.

 An EIOS must be sent prior to entering Electrical Idle if the current data rate is 2.5 GT/s or 8.0 GT/s. Two consecutive EIOSs must be sent prior to entering Electrical Idle if the current data rate is 5.0 GT/s.

 The DC common mode voltage is not required to be within specification.

 An Electrical Idle condition exists on the Lanes if an EIOS is received on any of the configured Lanes or Electrical Idle is detected/inferred as described in Section 4.2.4.3.

 - On entry to this substate following a successful speed negotiation (i.e., successful_speed_negotiation = 1b), an Electrical Idle condition may be inferred on the Receiver Lanes if a TS1 or TS2 Ordered Set has not been received in any configured Lane in a time interval specified in Table 4-11. (This covers the case where the Link is operational and both sides have successfully received TS Ordered Sets. Hence, a lack of a TS1 or TS2 Ordered Set in the specified interval can be interpreted as entry to Electrical Idle.)
 - Else on entry to this substate following an unsuccessful speed negotiation (i.e., successful_speed_negotiation = 0b) if an exit from Electrical Idle has not been detected at least once in any configured Lane in a time interval specified in Table 4-11. (This covers the case where at least one side is having trouble receiving TS Ordered Sets that

was transmitted by the other agent, and hence a lack of exit from Electrical Idle in a longer interval can be treated as equivalent to entry to Electrical Idle.)

- Next state is Recovery.RcvrLock after the Transmitter Lanes are no longer required to be in Electrical Idle as described in the condition above.
 - If this substate has been entered from Recovery.RcvrCfg following a successful speed change negotiation (i.e., successful_speed_negotiation = 1b), the new data rate is changed on all the configured Lanes to the highest common data rate advertised by both sides of the Link. The changed_speed_recovery variable is set to 1b.
 - Else if this substate is being entered for a second time since entering Recovery from L0 or L1 (i.e., changed_speed_recovery = 1b), the new data rate will be the data rate at which the LTSSM entered Recovery from L0 or L1. The changed_speed_recovery variable will be reset to 0b.
 - Else the new data rate will be 2.5 GT/s. The changed_speed_recovery variable remains reset at 0b.

 Note: This represents the case where the frequency of operation in L0 was greater than 2.5 GT/s and one side could not operate at that frequency and timed out in Recovery.RcvrLock the first time it entered that substate from L0 or L1.
- Next state is Detect after a 48 ms timeout.
 - Note: This transition is not possible under normal conditions.
- The directed_speed_change variable will be reset to 0b. The new data rate must be reflected in the Current Link Speed field of the Link Status register.
 - On a Link bandwidth change, if successful_speed_negotiation is set to 1b and the Autonomous Change bit (bit 6 of Symbol 4) in the eight consecutive TS2 Ordered Sets received while in Recovery.RcvrCfg is set to 1b or the speed change was initiated by the Downstream Port for autonomous reasons (non-reliability and not due to the setting of the Link Retrain bit), the Link Autonomous Bandwidth Status bit of the Link Status register is set to 1b.
 - Else: on a Link bandwidth change, the Link Bandwidth Management Status bit of the Link Status register is set to 1b.

4.2.6.4.4. Recovery.RcvrCfg

Transmitter sends TS2 Ordered Sets on all configured Lanes using the same Link and Lane numbers that were set after leaving Configuration. The speed_change bit (bit 7 of data rate identifier Symbol in TS2 Ordered Set) must be set to 1b if the directed_speed_change variable is already set to 1b. The N_FTS value in the transmitted TS2 Ordered Sets should reflect the number at the current data rate.

The Downstream Port must transmit EQ TS2 Ordered Sets (TS2 Ordered Sets with Symbol 6 bit 7 set to 1b) with the Transmitter Preset and Receiver Preset Hint fields set to the values specified by the Upstream Port Transmitter Preset and the Upstream Port Receiver Preset Hint fields of each Lane's Equalization Control register if all of the following conditions are satisfied:

a) The Downstream Port advertised 8.0 GT/s data rate support in Recovery.RcvrLock, and 8.0 GT/s data rate support has been advertised in the Configuration.Complete or Recovery.RcvrCfg substates by the Upstream Port since exiting the Detect state, and eight consecutive TS1 or TS2 Ordered Sets were received on any configured Lane prior to entry to this substate with speed_change bit set to 1b

b) The equalization_done_8GT_data_rate variable is 0b or if directed

An Upstream Port sets Request Equalization to 1b (Symbol 6 bit 7) if the current data rate is 8.0 GT/s and a problem was detected with the most recently executed equalization process. Otherwise, Request Equalization is set to 0b. An Upstream Port or Downstream Port may also set the Request Equalization and Quiesce Guarantee bits to 1b, if re-equalization is desired, based on the rules in Section 4.2.3.

The start_equalization_w_preset variable is reset to 0b upon entry to this substate.

- On entry to this substate, a Downstream Port must set the select_deemphasis variable equal to the Selectable De-emphasis field in the Link Control 2 register or adopt some implementation specific mechanism to set the select_deemphasis variable, including using the value requested by the Upstream Port in the eight consecutive TS1 Ordered Sets it received. A Downstream Port advertising 5.0 GT/s data rate support must set the Selectable De-emphasis bit (Symbol 4 bit 6) of the TS2 Ordered Sets it transmits identical to the select_deemphasis variable. An Upstream Port must set its Autonomous Change bit (Symbol 4 bit 6) to 1b in the TS2 Ordered Set if it intends to change the Link bandwidth for autonomous reasons.
 - For devices that support Link width upconfigure, it is recommended that the Electrical Idle detection circuitry be activated in the set of currently inactive Lanes in this substate, the Recovery.Idle substate, and Configuration.Linkwidth.Start substates, if the directed_speed_change variable is reset to 0b. This is done so that during a Link upconfigure, the side that does not initiate the upconfiguration does not miss the first EIEOS sent by the initiator during the Configuration.Linkwidth.Start substate.
- Next state is Recovery.Speed if all of the following conditions are true:
 - One of the following conditions is satisfied:
 i. Eight consecutive TS2 Ordered Sets are received on any configured Lane with identical data rate identifiers, identical values in Symbol 6, and the speed_change bit set to 1b and eight consecutive TS2 Ordered Sets are standard TS2 Ordered Sets if 8b/10b encoding is used
 ii. Eight consecutive EQ TS2 Ordered Sets are received on all configured Lanes with identical data rate identifiers, identical value in Symbol 6, and the speed_change bit set to 1b

iii. Eight consecutive EQ TS2 Ordered Sets are received on any configured Lane with identical data rate identifiers, identical value in Symbol 6, and the speed_change bit set to 1b and 1 ms has expired since the receipt of the eight consecutive EQ Ordered Sets on any configured Lane

- Either the current data rate is greater than 2.5 GT/s or greater than 2.5 GT/s data rate identifiers are set both in the transmitted and the (eight consecutive) received TS2 Ordered Sets
- For 8b/10b encoding, at least 32 TS2 Ordered Sets, without being interrupted by any intervening EIEOS, are transmitted with the speed_change bit set to 1b after receiving one TS2 Ordered Set with the speed_change bit set to 1b in the same configured Lane. For 128b/130b encoding, at least 128 TS2 Ordered Sets are transmitted with the speed_change bit set to 1b after receiving one TS2 Ordered Set with the speed_change bit set to 1b in the same configured Lane.

The data rate(s) advertised on the received eight consecutive TS2 Ordered Sets with the speed_change bit set is noted as the data rate(s) that can be supported by the other Port. The Autonomous Change bit (Symbol 4 bit 6) in these received eight consecutive TS2 Ordered Sets is noted by the Downstream Port for possible logging in the Link Status register in Recovery.Speed substate. Upstream Ports must register the Selectable De-emphasis bit (bit 6 of Symbol 4) advertised in these eight consecutive TS2 Ordered Sets in the select_deemphasis variable. The new speed to change to in Recovery.Speed is the highest data rate that can be supported by both Ports on the Link. For an Upstream Port, if these eight consecutive TS2 Ordered Sets are EQ TS2 Ordered Sets, it must set the start_equalization_w_preset variable to 1b and update the Upstream Port Transmitter Preset and Upstream Port Receiver Preset Hint fields of the Lane Equalization Control register with the values received in the eight consecutive EQ TS2 Ordered Sets. Any configured Lanes which do not receive EQ TS2 Ordered Sets meeting this criteria will use implementation dependent preset values when first operating at 8.0 GT/s prior to performing link equalization. A Downstream Port must set the start_equalization_w_preset variable to 1b if the equalization_done_8GT_data_rate variable is 0b or if directed. The variable successful_speed_negotiation is set to 1b. Note that if the Link is already operating at the highest data rate supported by both Ports, Recovery.Speed is executed but the data rate is not changed. If 128b/130b encoding is used, and the Request Equalization bit is set to 1b in the eight consecutive TS2 Ordered Sets, the device must note it along with the Quiesce Guarantee bit and follow the rules in Section 4.2.3.

- Next state is Recovery.Idle if the following two conditions are both true:
 - Eight consecutive TS2 Ordered Sets are received on all configured Lanes with the same Link and Lane number that match what is being transmitted on those same Lanes with identical data rate identifiers within each Lane and one of the following two sub-conditions are true:
 - the speed_change bit is 0b in the received eight consecutive TS2 Ordered Sets
 - current data rate is 2.5 GT/s and either no 5.0 GT/s, or higher, data rate identifiers are set in the received eight consecutive TS2 Ordered Sets, or no 5.0 GT/s, or higher, data rate identifiers are being transmitted in the TS2 Ordered Sets

- 16 TS2 Ordered Sets are sent after receiving one TS2 Ordered Set without being interrupted by any intervening EIEOS. The changed_speed_recovery variable and the directed_speed_change variable are reset to 0b on entry to Recovery.Idle.
- If the N_FTS value was changed, the new value must be used for future L0s states.
- When using 8b/10b encoding, Lane-to-Lane de-skew must be completed before leaving Recovery.RcvrCfg.
- The device must note the data rate identifier advertised on any configured Lane in the eight consecutive TS2 Ordered Sets described in this state transition. This will override any previously recorded value.
- When using 128b/130b encoding and if the Request Equalization bit is set to 1b in the eight consecutive TS2 Ordered Sets, the device must note it and follow the rules in Section 4.2.3.

❑ Next state is Configuration if eight consecutive TS1 Ordered Sets are received on any configured Lanes with Link or Lane numbers that do not match what is being transmitted on those same Lanes and 16 TS2 Ordered Sets are sent after receiving one TS1 Ordered Set and one of the following two conditions apply:

- the speed_change bit is 0b on the received TS1 Ordered Sets
- current data rate is 2.5 GT/s and either no 5.0 GT/s, or higher, data rate identifiers are set in the received eight consecutive TS1 Ordered Sets, or no 5.0 GT/s, or higher, data rate identifiers are being transmitted in the TS2 Ordered Sets

The changed_speed_recovery variable and the directed_speed_change variable are reset to 0b if the LTSSM transitions to Configuration.

- If the N_FTS value was changed, the new value must be used for future L0s states.

❑ Next state is Recovery.Speed if the speed of operation has changed to a mutually negotiated data rate since entering Recovery from L0 or L1 (i.e., changed_speed_recovery = 1b) and an EIOS has been detected or an Electrical Idle condition has been inferred/detected on any of the configured Lanes and no configured Lane received a TS2 Ordered Set since entering this substate (Recovery.RcvrCfg). The new data rate to operate after leaving Recovery.Speed will be reverted back to the speed of operation during entry to Recovery from L0 or L1.

As described in Section 4.2.4.3, an Electrical Idle condition may be inferred if a TS1 or TS2 Ordered Set has not been received in a time interval specified in Table 4-11.

❑ Next state is Recovery.Speed if the speed of operation has not changed to a mutually negotiated data rate since entering Recovery from L0 or L1 (i.e., changed_speed_recovery = 0b) and the current speed of operation is greater than 2.5 GT/s and an EIOS has been detected or an Electrical Idle condition has been detected/inferred on any of the configured Lanes and no configured Lane received a TS2 Ordered Set since entering this substate (Recovery.RcvrCfg). The new data rate to operate after leaving Recovery.Speed will be 2.5 GT/s.

As described in Section 4.2.4.3, an Electrical Idle condition may be inferred if a TS1 or TS2 Ordered Set has not been received in a time interval specified in Table 4-11.

Note: This transition implies that the other side was unable to achieve Symbol lock or Block alignment at the speed with which it was operating. Hence both sides will go back to the

2.5 GT/s speed of operation and neither device will attempt to change the speed again without exiting Recovery state. It should also be noted that even though a speed change is involved here, the changed_speed_recovery will be 0b.

- After a 48 ms timeout;
 - The next state is Detect if the current data rate is 2.5 GT/s or 5.0 GT/s.
 - The next state is Recovery.Idle if the idle_to_rlock_transitioned variable is less than FFh and the current data rate is 8.0 GT/s.
 i. The changed_speed_recovery variable and the directed_speed_change variable are reset to 0b on entry to Recovery.Idle.
 - Else the next state is Detect.

4.2.6.4.5. Recovery.Idle

- Next state is Disabled if directed.
 - Note: "if directed" applies to a Downstream or optional crosslink Port that is instructed by a higher Layer to assert the Disable Link bit (TS1 and TS2) on the Link.
- Next state is Hot Reset if directed.
 - Note: "if directed" applies to a Downstream or optional crosslink Port that is instructed by a higher Layer to assert the Hot Reset bit (TS1 and TS2) on the Link.
- Next state is Configuration if directed.
 - Note: "if directed" applies to a Port that is instructed by a higher Layer to optionally re-configure the Link (i.e., different width Link).
- Next state is Loopback if directed to this state, and the Transmitter is capable of being a loopback master, which is determined by implementation specific means.
 - Note: "if directed" applies to a Port that is instructed by a higher Layer to assert the Loopback bit (TS1 and TS2) on the Link.
- Next state is Disabled immediately after any configured Lane has the Disable Link bit asserted in two consecutively received TS1 Ordered Sets.
 - Note: This is behavior only applicable to Upstream and optional crosslink Ports.
- Next state is Hot Reset immediately after any configured Lane has the Hot Reset bit asserted in two consecutive TS1 Ordered Sets.
 - Note: This is behavior only applicable to Upstream and optional crosslink Ports.
- Next state is Configuration if two consecutive TS1 Ordered Sets are received on any configured Lane with a Lane number set to PAD.
 - Note: A Port that optionally transitions to Configuration to change the Link configuration is guaranteed to send Lane numbers set to PAD on all Lanes.
 - Note: It is recommended that the LTSSM initiate a Link width up/downsizing using this transition to reduce the time it takes to change the Link width.

- Next state is Loopback if any configured Lane has the Loopback bit asserted in two consecutive TS1 Ordered Sets.
 - Note: The device receiving the Ordered Set with the Loopback bit set becomes the loopback slave.
- When using 8b/10b encoding, the Transmitter sends Idle data on all configured Lanes. When using 128b/130b encoding, the Transmitter sends one SDS Ordered Set on all configured Lanes to start a Data Stream and then sends Idle data Symbols on all configured Lanes. Note: The first Idle data Symbol transmitted on Lane 0 is the first Symbol of the Data Stream.
 - If directed to other states, Idle Symbols do not have to be sent, and must not be sent with 128b/130b encoding, before transitioning to the other states (i.e., Disabled, Hot Reset, Configuration, or Loopback)



IMPLEMENTATION NOTE

EDS Usage

In 128b/130b encoding, on transition to Configuration or Loopback or Hot Reset or Disabled, an EDS must be sent if a Data Stream is active (i.e., an SDS Ordered Set has been sent). It is possible that the side that is not initiating Link Upconfigure has already transmitted SDS and transmitting Data Stream (Logical IDL) when it receives the TS1 Ordered Sets. In that situation, it will send EDS in the set of Lanes that are active before sending the TS1 Ordered Sets in Configuration.

- When using 8b/10b encoding, next state is L0 if eight consecutive Symbol Times of Idle data are received on all configured Lanes and 16 Idle data Symbols are sent after receiving one Idle data Symbol.
 - If software has written a 1b to the Retrain Link bit in the Link Control register since the last transition to L0 from Recovery or Configuration, the Downstream Port must set the Link Bandwidth Management Status bit of the Link Status register to 1b.
- When using 128b/130b encoding, next state is L0 if eight consecutive Symbol Times of Idle data are received on all configured Lanes, 16 Idle data Symbols are sent after receiving one Idle data Symbol, and this state was not entered by a timeout from Recovery.RcvrCfg
 - The Idle data Symbols must be received in Data Blocks.
 - Lane-to-Lane de-skew must be completed before Data Stream processing starts.
 - If software has written a 1b to the Retrain Link bit in the Link Control register since the last transition to L0 from Recovery or Configuration, the Downstream Port must set the Link Bandwidth Management Status bit of the Link Status register to 1b.
 - The idle_to_rlock_transitioned variable is reset to 00h on transition to L0.
- Otherwise, after a 2 ms timeout:
 - If the idle_to_rlock_transitioned variable is less than FFh, the next state is Recovery.RcvrLock.
 - If the data rate is 8.0 GT/s, the idle_to_rlock_transitioned variable is incremented by 1b upon transitioning to Recovery.RcvrLock.

- If the data rate is 5.0 GT/s (or, if supported in 2.5 GT/s), the idle_to_rlock_transitioned variable is set to FFh, upon transitioning to Recovery.RcvrLock.

o Else the next state is Detect

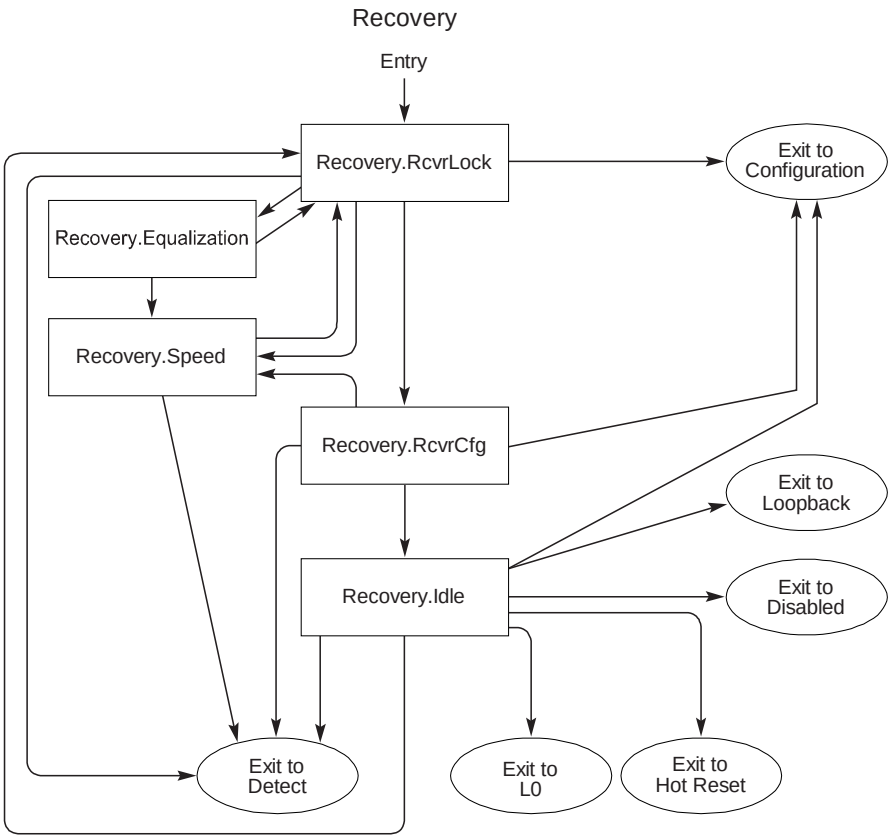


Figure 4-26: Recovery Substate Machine

4.2.6.5. L0

This is the normal operational state.

- LinkUp = 1b (status is set true).
 - On receipt of an STP or SDP Symbol, the idle_to_rlock_transitioned variable is reset to 00h.
- Next state is Recovery if either of the two conditions are satisfied (i) if directed to change speed (directed_speed_change variable = 1b) by a higher layer when both sides support greater than 2.5 GT/s speeds and the Link is in DL_Active state, or (ii) if directed to change speed (directed_speed_change variable = 1b) by a higher layer when both sides support 8.0 GT/s data rate to perform Transmitter Equalization at 8.0 GT/s data rate. The changed_speed_recovery bit is reset to 0b.
 - For an Upstream Port, the directed_speed_change variable must not be set to 1b if it has never recorded greater than 2.5 GT/s data rate support advertised in Configuration.Complete or Recovery.RcvrCfg substates by the Downstream Port since exiting the Detect state.
 - For a Downstream Port , the directed_speed_change variable must not be set to 1b if it has never recorded greater than 2.5 GT/s data rate support advertised in Configuration.Complete or Recovery.RcvrCfg substates by the Upstream Port since exiting the Detect state. If greater than 2.5 GT/s data rate support has been noted, the Downstream Port must set the directed_speed_change variable to 1b if the Retrain Link bit of the Link Control register is set to 1b and the Target Link Speed field in the Link Control 2 register is not equal to the current Link speed.
 - A Port supporting greater than 2.5 GT/s data rates must participate in the speed change even if the Link is not in DL_Active state if it is requested by the other side through the TS Ordered Sets.
- Next state is Recovery if directed to change Link width.
 - The upper layer must not direct a Port to increase the Link width if the other Port did not advertise the capability to upconfigure the Link width during the Configuration state or if the Link is currently operating at the maximum possible width it negotiated on initial entry to the L0 state.
 - Normally, the upper layer will not reduce width if upconfigure_capable is reset to 0b other than for reliability reasons, since the Link will not be able to go back to the original width if upconfigure_capable is 0b. A Port must not initiate reducing the Link width for reasons other than reliability if the Hardware Autonomous Width Disable bit in the Link Control register is set to 1b.
 - The decision to initiate an increase or decrease in the Link width, as allowed by the specification, is implementation specific.
- Next state is Recovery if a TS1 or TS2 Ordered Set is received on any configured Lane or an EIEOS is received on any configured Lane in 128b/130b encoding.
- Next state is Recovery if directed to this state. If Electrical Idle is detected/inferred on all Lanes without receiving an EIOS on any Lane, the Port may transition to the Recovery state or may

remain in L0. In the event that the Port is in L0 and the Electrical Idle condition occurs without receiving an EIOS, errors may occur and the Port may be directed to transition to Recovery.

- As described in Section 4.2.4.3, an Electrical Idle condition may be inferred on all Lanes under any one of the following conditions: (i) absence of a Flow Control Update DLLP in any 128 µs window, (ii) absence of a SKP Ordered Set in any of the configured Lanes in any 128 µs window, or (iii) absence of either a Flow Control Update DLLP or a SKP Ordered Set in any of the configured Lanes in any 128 µs window.
- Note: "if directed" applies to a Port that is instructed by a higher Layer to transition to Recovery including the Retrain Link bit in the Link Control register being set.
- The Transmitter may complete any TLP or DLLP in progress.

❑ Next state is L0s for only the Transmitter if directed to this state and the Transmitter implements L0s. See Section 4.2.6.6.2.

- Note: "if directed" applies to a Port that is instructed by a higher Layer to initiate L0s (see Section 5.4.1.1.1).
- Note: This is a point where the TX and RX may diverge into different LTSSM states.

❑ Next state is L0s for only the Receiver if an EIOS is received on any Lane, the Receiver implements L0s, and the Port is not directed to L1 or L2 states by any higher layers. See Section 4.2.6.6.1.

- Note: This is a point where the TX and RX may diverge into different LTSSM states.

❑ Next state is Recovery if an EIOS is received on any Lane, the Receiver does not implement L0s, and the Port is not directed to L1 or L2 states by any higher layers. See Section 4.2.6.6.1.

❑ Next state is L1:

(i) If directed

and

(ii) an EIOS is received on any Lane

and

(iii) an EIOS is transmitted on all Lanes if the current Link speed is 2.5 GT/s or 8.0 GT/s or two consecutive EIOSs are transmitted on all Lanes if the current Link speed is 5.0 GT/s.

- Note: "if directed" is defined as both ends of the Link having agreed to enter L1 immediately after the condition of both the receipt and transmission of the EIOS(s) is met. A transition to L1 can be initiated by PCI-PM (see Section 5.3.2.1) or by ASPM (see Section 5.4.1.2.1).
- Note: When directed by a higher Layer one side of the Link always initiates and exits to L1 by transmitting the EIOS(s) on all Lanes, followed by a transition to Electrical Idle.[54] The same Port then waits for the receipt of an EIOS on any Lane, and then immediately

[54] The common mode being driven must meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX\text{-}CM\text{-}DC\text{-}ACTIVE\text{-}IDLE\text{-}DELTA}$) specification (see Table 4-18).

transitions to L1. Conversely, the side of the Link that first receives the EIOS(s) on any Lane must send an EIOS on all Lanes and immediately transition to L1.

- Next state is L2:

 (i) If directed

 and

 (ii) an EIOS is received on any Lane

 (iii) an EIOS is transmitted on all Lanes if the current Link speed is 2.5 GT/s or 8.0 GT/s or two consecutive EIOSs are transmitted on all Lanes if the current Link speed is 5.0 GT/s.

 - Note: "if directed" is defined as both ends of the Link having agreed to enter L2 immediately after the condition of both the receipt and transmission of the EIOS(s) is met (see Section 5.3.2.3 for more details).
 - Note: When directed by a higher Layer, one side of the Link always initiates and exits to L2 by transmitting EIOS on all Lanes followed by a transition to Electrical Idle.[55] The same Port then waits for the receipt of EIOS on any Lane, and then immediately transitions to L2. Conversely, the side of the Link that first receives an EIOS on any Lane must send an EIOS on all Lanes and immediately transition to L2.

4.2.6.6. L0s

The L0s substate machine is shown in Figure 4-27.

4.2.6.6.1. Receiver L0s

A Receiver must implement L0s if its Port advertises support for L0s, as indicated by the ASPM Support field in the Link Capability register. It is permitted for a Receiver to implement L0s even if its Port does not advertise support for L0s.

[55] The common mode being driven does not need to meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle (VTX-CM-DC-ACTIVE-IDLE-DELTA) specification (see Table 4 18).

4.2.6.6.1.1. Rx_L0s.Entry

- Next state is Rx_L0s.Idle after a $T_{TX\text{-}IDLE\text{-}MIN}$ (Table 4-18) timeout
 - Note: This is the minimum time the Transmitter must be in an Electrical Idle condition.

4.2.6.6.1.2. Rx_L0s.Idle

- Next state is Rx_L0s.FTS if the Receiver detects an exit from Electrical Idle on any Lane of the configured Link.
- Next state is Rx_L0s.FTS after a 100 ms timeout if the current data rate is 8.0 GT/s or higher and the Port's Receivers do not meet the $Z_{RX\text{-}DC}$ specification for 2.5 GT/s (see Table 4-24). All Ports are permitted to implement the timeout and transition to Rx_L0s.FTS when the data rate is 8.0 GT/s or higher.

Comment [VS128]:
ECR: 8.0 GT/s Receiver Impedance

4.2.6.6.1.3. Rx_L0s.FTS

- The next state is L0 if a SKP Ordered Set is received in 8b/10b encoding or the SDS Ordered Set is received for 128b/130b encoding on all configured Lanes of the Link.
 - The Receiver must be able to accept valid data immediately after the SKP Ordered Set for 8b/10b encoding.
 - The Receiver must be able to accept valid data immediately after the SDS Ordered Set for 128b/130b encoding.
 - Lane-to-Lane de-skew must be completed before leaving Rx_L0s.FTS.
- Otherwise, next state is Recovery after the N_FTS timeout.
 - When using 8b/10b encoding: The N_FTS timeout shall be no shorter than 40*[N_FTS+3] * UI (The 3 * 40 UI is derived from six Symbols to cover a maximum SKP Ordered Set + four Symbols for a possible extra FTS + 2 Symbols of design margin), and no longer than twice this amount. When the extended synch bit is set the Receiver N_FTS timeout must be adjusted to no shorter than 40* [2048] * UI (2048 FTSs) and no longer than 40* [4096] * UI (4096 FTSs). Implementations must take into account the worst case Lane to Lane skew, their design margins, as well as the four to eight consecutive EIE Symbols in speeds other than 2.5 GT/s when choosing the appropriate timeout value within the specification's defined range.
 - When using 128b/130b encoding: The N_FTS timeout shall be no shorter than 130*[N_FTS+5+ 12 + Floor(N_FTS/32)] * UI and no longer than twice this amount. The 5 + Floor(N_FTS/32) accounts for the first EIEOS, the last EIEOS, the SDS, the periodic EIEOS and an additional EIEOS in case an implementation chooses to send two EIEOS followed by an SDS when N_FTS is divisible by 32. The 12 is there to account for the number of SKP Ordered Sets that will be transmitted if extended synch bit is set. When the extended synch bit is set, the timeout should be the same as the normal case with N_FTS equal to 4096.
 - The Transmitter must also transition to Recovery, but is permitted to complete any TLP or DLLP in progress.

- It is recommended that the N_FTS field be increased when transitioning to Recovery to prevent future transitions to Recovery from Rx_L0s.FTS.

4.2.6.6.2. Transmitter L0s

A Transmitter must implement L0s if its Port advertises support for L0s, as indicated by the ASPM Support field in the Link Capability register. It is permitted for a Transmitter to implement L0s even if its Port does not advertise support for L0s.

4.2.6.6.2.1. Tx_L0s.Entry

- ❑ Transmitter sends one EIOS if the current data rate is 2.5 GT/s or 8.0 GT/s or two consecutive EIOSs if the current data rate is 5.0 GT/s and enters Electrical Idle.
 - The DC common mode voltage must be within specification by $T_{TX\text{-}IDLE\text{-}SET\text{-}TO\text{-}IDLE}$.[56]
- ❑ Next state is Tx_L0s.Idle after a $T_{TX\text{-}IDLE\text{-}MIN}$ (Table 4-18) timeout.

4.2.6.6.2.2. Tx_L0s.Idle

- ❑ Next state is Tx_L0s.FTS if directed.



IMPLEMENTATION NOTE

Increase of N_FTS Due to Timeout in Rx_L0s.FTS

The Transmitter sends the N_FTS fast training sequences by going through Tx_L0s.FTS substates to enable the Receiver to reacquire its bit and Symbol lock or Block alignment. In the absence of the N_FTS fast training sequence, the Receiver will timeout in Rx_L0s.FTS substate and may increase the N_FTS number it advertises in the Recovery state.

[56] The common mode being driven must meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX\text{-}CM\text{-}DC\text{-}ACTIVE\text{-}IDLE\text{-}DELTA}$) specification (see Table 4-18).

4.2.6.6.2.3. Tx_L0s.FTS

- Transmitter must send~~s~~ N_FTS Fast Training Sequences on all configured Lanes.
 - Four to eight EIE Symbols must be sent prior to transmitting the N_FTS (or 4096 if Extended Sync = 1) number of FTS in 5.0 GT/s data rates. An EIEOS must be sent prior to transmitting the N_FTS (or 4096 if Extended Sync = 1) number of FTS with 128b/130b encoding. In 2.5 GT/s speed, up to one full FTS may be sent before the N_FTS (or 4096 if Extended Sync = 1) number of FTSs are sent.
 - ~~No~~ SKP Ordered Sets must not~~can~~ be inserted before all FTSs as defined by the agreed upon N_FTS parameter are transmitted.
 - If the Extended Synch bit is set, the Transmitter must send~~s~~ 4096 Fast Training Sequences , inserting SKP Ordered Sets according to the requirements in Section 4.2.4.5.
- When using 8b/10b encoding, the Transmitter must send~~s~~ a single SKP Ordered Set on all configured Lanes.
- When using 128b/130b encoding, the Transmitter must send~~s~~ one EIEOS followed by one SDS Ordered Set on all configured Lanes. Note: The first Symbol transmitted on Lane 0 after the SDS Ordered Set is the first Symbol of the Data Stream.
- Next state must be~~is~~ L0, after completing the above required transmissions.

Comment [VS129]: Errata B1

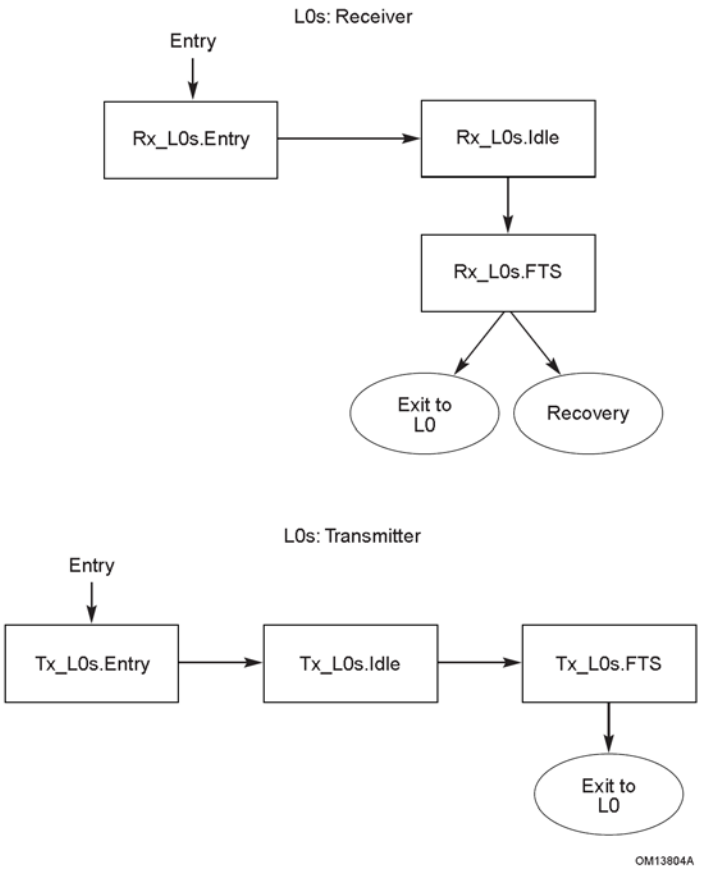


Figure 4-27: L0s Substate Machine

4.2.6.7. L1

The L1 substate machine is shown in Figure 4-28.

4.2.6.7.1. L1.Entry

- ❑ All configured Transmitters are in Electrical Idle.
 - o The DC common mode voltage must be within specification by $T_{TX-IDLE-SET-TO-IDLE}$.
- ❑ The next state is L1.Idle after a $T_{TX-IDLE-MIN}$ (Table 4-18) timeout.
 - o Note: This guarantees that the Transmitter has established the Electrical Idle condition.

4.2.6.7.2. L1.Idle

- ❑ Transmitter remains in Electrical Idle.
- ❑ The DC common mode voltage must be within specification, except as allowed by L1 PM Substates, when applicable.[57]
- ❑ A substate of L1 is entered when the conditions for L1 PM Substates are satisfied (see Section 5.5).
 - o The L1 PM Substate must be L1.0 when L1.Idle is entered or exited.
- ❑ Next state is Recovery if ~~any Receiver detects~~ exit from Electrical Idle is detected on any Lane of a configured Link, or directed after remaining in this substate for a minimum of 40 ns in speeds other than 2.5 GT/s.
 - o Ports are not required to arm the Electrical Idle exit detectors on all Lanes of the Link.
 - o Note: A minimum stay of 40 ns is required in this substate in speeds other than 2.5 GT/s to account for the delay in the logic levels to arm the Electrical Idle detection circuitry in case the Link enters L1 and immediately exits the L1 state.
 - o A Port is allowed to set the directed_speed_change variable to 1b following identical rules described in L0 for setting this variable. When making such a transition, the changed_speed_recovery variable must be reset to 0b. A Port may also go through Recovery back to L0 and then set the directed_speed_change variable to 1b on the transition from L0 to Recovery.
 - o A Port is also allowed to enter Recovery from L1 if directed to change the Link width. The Port must follow identical rules for changing the Link width as described in the L0 state.
- ❑ Next state is Recovery after a 100 ms timeout if the current data rate is 8.0 GT/s or higher and the Port's Receivers do not meet the Z_{RX-DC} specification for 2.5 GT/s (see Table 4-24). All

Comment [VS130]: ECN: L1 PM Substates with CLKREQ

Comment [VS131]: ECN: L1 PM Substates with CLKREQ

[57] The common mode being driven must meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$) specification (see Table 4-18).

Ports are permitted, but not encouraged, to implement the timeout and transition to Recovery when the data rate is 8.0 GT/s or higher.

- This timeout is not affected by the L1 PM Substates mechanism.

IMPLEMENTATION NOTE

100 ms Timeout in L1

Ports that meet the Z_{RX-DC} specification for 2.5 GT/s while in the L1.Idle state and are therefore not required to implement the 100 ms timeout and transition to Recovery should avoid implementing it, since it will reduce the power savings expected from the L1 state.

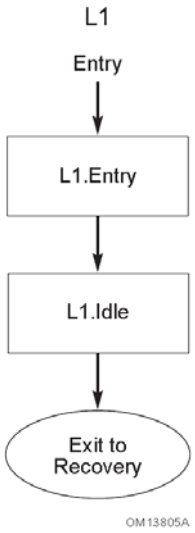


Figure 4-28: L1 Substate Machine

4.2.6.8. L2

The L2 substate machine is shown in Figure 4-29.

4.2.6.8.1. L2.Idle

- ~~All Rx termination must remain enabled in low impedance.~~
- All Receivers must meet the Z_{RX-DC} specification for 2.5 GT/s within 1 ms (see Table 4-24).
- All configured Transmitters must remain in Electrical Idle for a minimum time of $T_{TX-IDLE-MIN}$ (Table 4-18).
 - The DC common mode voltage does not have to be within specification.

Comment [VS132]: ECR: 8.0 GT/s Receiver Impedance

Comment [VS133]: ECN: L1 PM Substates with CLKREQ

Formatted: Font color: Black

Comment [VS134]: ECR: 8.0 GT/s Receiver Impedance

Comment [VS135]: ECR: 8.0 GT/s Receiver Impedance

Comment [VS136]: ECR: 8.0 GT/s Receiver Impedance

- The Receiver needs to wait a minimum of $T_{TX-IDLE-MIN}$ to start looking for Electrical Idle Exit.

- For Downstream Lanes:
 - For a Root Port, the next state is Detect if a Beacon is received on at least Lane 0 or if directed.
 - Main power must be restored before entering Detect.
 - Note: "if directed" is defined as a higher layer decides to exit to Detect.
 - For a Switch, if a Beacon is received on at least Lane 0, the Upstream Port must transition to L2.TransmitWake.
- For Upstream Lanes:
 - The next state is Detect if Electrical Idle Exit is detected on any predetermined set of Lanes.
 - The predetermined set of Lanes must include but is not limited to any Lane which has the potential of negotiating to Lane 0 of a Link. For multi-Lane Links the number of Lanes in the predetermined set must be greater than or equal to two.
 - A Switch must transition any Downstream Lanes to Detect.
 - Next state is L2.TransmitWake for an Upstream Port if directed to transmit a Beacon.
 - Note: Beacons may only be transmitted on Upstream Ports in the direction of the Root Complex.

4.2.6.8.2. L2.TransmitWake

This state only applies to Upstream Ports.

- Transmit the Beacon on at least Lane 0 (see Section 4.3.5.7).
- Next state is Detect if Electrical Idle exit is detected on any Upstream Port's Receiver that is in the direction of the Root Complex.
 - Note: Power is guaranteed to be restored when Upstream Receivers see Electrical Idle exited, but it may also be restored prior to Electrical Idle being exited.

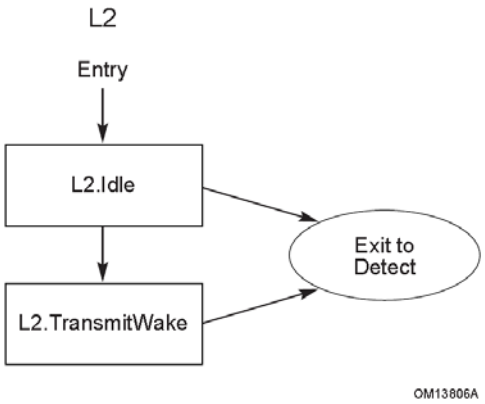


Figure 4-29: L2 Substate Machine

4.2.6.9. Disabled

- All Lanes transmit 16 to 32 TS1 Ordered Sets with the Disable Link bit asserted and then transition to Electrical Idle.
 - The EIOS (one EIOS if the current Link speed is 2.5 GT/s or 8.0 GT/s and two consecutive EIOSs if the current Link speed is 5.0 GT/s) must be sent prior to entering Electrical Idle.
 - The DC common mode voltage does not have to be within specification.[58]
- If an EIOS was transmitted (one if the current Link speed is 2.5 GT/s or 8.0 GT/s and two consecutive ones if the current Link speed is 5.0 GT/s) and an EIOS was received on any Lane (even while transmitting TS1 with the Disable Link bit asserted), then:
 - LinkUp = 0b (False)
 - At this point, the Lanes are considered Disabled.
 - For Upstream Ports: All Receivers must meet the Z_{RX-DC} specification for 2.5 GT/s within 1 ms (see Table 4-24).
 - For Upstream Ports: The next state is Detect when Electrical Idle Exit is detected at the Receiver.
- For Downstream Ports: The next state is Detect when directed (e.g., when the Link Disable bit is reset to 0b by software).
- For Upstream Ports: If no EIOS is received after a 2 ms timeout, the next state is Detect.

Comment [VS137]: ECR: 8.0 GT/s Receiver Impedance

[58] The common mode being driven does need to meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$) specification (see Table 4-18).

4.2.6.10. Loopback

The Loopback substate machine is shown in Figure 4-30.

4.2.6.10.1. Loopback.Entry

- LinkUp = 0b (False)
- The Link and Lane numbers received in the TS1 or TS2 Ordered Sets are ignored by the Receiver while in this substate.
- Loopback master requirements:
 - If Loopback.Entry was entered from Configuration.Linkwidth.Start, determine the highest common data rate of the data rates supported by the master and the data rates received in two consecutive TS1 or TS2 Ordered Sets on any active Lane at the time the transition to Loopback.Entry occurred. If the current data rate is not the highest common data rate:
 - Transmit 16 consecutive TS1 Ordered Sets with the Loopback bit asserted, followed by one Electrical Idle Ordered Set when the current data rate is 2.5 GT/s or 8.0 GT/s or two Electrical Idle Ordered Sets when the current data rate is 5.0 GT/s, and then transition to Electrical Idle for 1 ms. During the period of Electrical Idle, change the data rate to the highest common data rate.
 - If the highest common data rate is 5.0 GT/s, the slave's transmitter de-emphasis is controlled by setting the Selectable De-emphasis bit of the transmitted TS1 Ordered Sets to the desired value (1b = -3.5 dB, 0b = -6 dB).
 - For data rates of 5.0 GT/s and above, the master is permitted to choose its own transmitter settings in an implementation-specific manner, regardless of the settings it transmitted to the slave.
 - Note: If Loopback is entered after LinkUp has been set to 1b, it is possible for one Port to enter Loopback from Recovery and the other to enter Loopback from Configuration. The Port that entered from Configuration might attempt to change data rate while the other Port does not. If this occurs, the results are undefined. The test set-up must avoid such conflicting directed clauses.
 - Transmit TS1 Ordered Sets with the Loopback bit asserted.
 - The master is also permitted to assert the Compliance Receive bit of TS1 Ordered Sets transmitted in Loopback.Entry, including those transmitted before a data rate change. If it asserts the Compliance Receive bit, it must not de-assert it again while in the Loopback.Entry state. This usage model might be helpful for test and validation purposes when one or both Ports have difficulty obtaining bit lock, Symbol lock, or Block alignment after a data rate change. The ability to set the Compliance Receive bit is implementation-specific.
 - Next state is Loopback.Active after 2 ms if the Compliance Receive bit of the transmitted TS1 Ordered Sets is asserted.

- Next state is Loopback.Active if the Compliance Receive bit of the transmitted TS1 Ordered Sets is deasserted and an implementation-specific set of Lanes receive two consecutive TS1 Ordered Sets with the Loopback bit asserted.

 If the data rate was changed, the master must take into account the amount of time the slave can be in Electrical Idle and transmit a sufficient number of TS1 Ordered Sets for the slave to acquire Symbol lock or Block alignment before proceeding to Loopback.Active.



IMPLEMENTATION NOTE

Lane Numbering with 128b/130b Encoding in Loopback

If the current data rate uses 128b/130b encoding and Lane numbers have not been negotiated, it is possible that the master and slave will not be able to decode received information because their Lanes are using different scrambling LFSR seed values (since the LFSR seed values are determined by the Lane numbers). This situation can be avoided by allowing the master and slave to negotiate Lane numbers before directing the master to Loopback, directing the master to assert the Compliance Receive bit during Loopback.Entry, or by using some other method of ensuring that the LFSR seed values match.

- Next state is Loopback.Exit after an implementation-specific timeout of less than 100 ms.

❑ Loopback slave requirements:

- If Loopback.Entry was entered from Configuration.Linkwidth.Start, determine the highest common data rate of the data rates supported by the slave and the data rates received in the two consecutive TS1 Ordered Sets that directed the slave to this state. If the current data rate is not the highest common data rate:
 - Transmit one Electrical Idle Ordered Set when the current data rate is 2.5 GT/s or 8.0 GT/s or two Electrical Idle Ordered Sets when the current data rate is 5.0 GT/s, and then transition to Electrical Idle for 2 ms. During the period of Electrical Idle, change the data rate to the highest common data rate.
 - If operating in full swing mode and the highest common data rate is 5.0 GT/s, set the transmitter's de-emphasis to the setting specified by the Selectable De-emphasis bit received in the TS1 Ordered Sets that directed the slave to this state. The de-emphasis is -3.5 dB if the Selectable De-emphasis bit was 1b, and it is -6 dB if the Selectable De-emphasis bit was 0b.
 - If the highest common data rate is 8.0 GT/s and EQ TS1 Ordered Sets directed the slave to this state, set the transmitter to the settings specified by the Preset field of the EQ TS1 Ordered Sets. See Section 4.2.3.2. If the highest common data rate is 8.0 GT/s but standard TS1 Ordered Sets directed the slave to this state, the slave is permitted to use its default transmitter settings.
- Next state is Loopback.Active if the Compliance Receive bit of the TS1 Ordered Sets that directed the slave to this state was asserted.

Comment [VS138]: Errata B24 (part of 2.1 Errata B68)

- The slave's transmitter does not need to transition to transmitting looped-back data on any boundary, and it is permitted to truncate any Ordered Set in progress.

- Else, the slave transmits TS1 Ordered Sets with Link and Lane numbers set to PAD.
 - Next state is Loopback.Active if the data rate is 2.5 GT/s or 5.0 GT/s and Symbol lock is obtained on all active Lanes.
 - Next state is Loopback.Active if the data rate is 8.0 GT/s and two consecutive TS1 Ordered Sets are received on all active Lanes. The equalization settings specified by the received TS1 Ordered Sets must be evaluated and applied to the transmitter if the value of the EC field is appropriate for the slave's Port direction (10b or 11b) and the requested setting is a preset or a set of valid coefficients. (Note: This is the equivalent behavior for the Recovery.Equalization state.) Optionally, the slave can accept both EC field values. If the settings are applied, they must take effect within 500 ns of being received, and they must not cause the transmitter to violate any electrical specification for more than 1 ns. Unlike Recovery.Equalization, the new settings are not reflected in the TS1 Ordered Sets that the slave transmits.
 - When using 8b/10b encoding, the slave's transmitter must transition to transmitting looped-back data on a Symbol boundary, but it is permitted to truncate any Ordered Set in progress. When using 128b/ 130b encoding, the slave's transmitter does not need to transition to transmitting looped-back data on any boundary, and is permitted to truncate any Ordered Set in progress.

4.2.6.10.2. Loopback.Active

- The loopback master must send valid encoded data. The loopback master should avoid sending EIOS as data until it wants to exit Loopback.
- A loopback slave is required to retransmit the received encoded information as received, with the polarity inversion determined in Polling applied, while continuing to perform clock tolerance compensation:
 - SKPs must be added or deleted on a per Lane basis as outlined in Section 4.2.7 with the exception that SKPs do not have to be simultaneously added or removed across Lanes of a configured Link.
 - For 8b/10b encoding, if a SKP Ordered Set retransmission requires adding a SKP Symbol to accommodate timing tolerance correction, the SKP Symbol is inserted in the retransmitted Symbol stream anywhere adjacent to a SKP Symbol in the SKP Ordered Set following the COM Symbol. The inserted SKP Symbol must be of the same disparity as the received SKPs Symbol(s) in the SKP Ordered Set.
 - For 8b/10b encoding, if a SKP Ordered Set retransmission requires dropping a SKP Symbol to accommodate timing tolerance correction, the SKP Symbol is simply not retransmitted.

- For 128b/130b encoding, if a SKP Ordered Set retransmission requires adding SKP Symbols to accommodate timing tolerance correction, four SKP Symbols are inserted in the retransmitted Symbol stream prior to the SKP_END Symbol in the SKP Ordered Set.
- For 128b/130b encoding, if a SKP Ordered Set retransmission requires dropping SKP Symbols to accommodate timing tolerance correction, four SKP Symbols prior to the SKP_END Symbol in the SKP Ordered Set are simply not retransmitted.

- No modifications of the received encoded data (except for polarity inversion determined in Polling) are allowed by the loopback slave even if it is determined to be an invalid encoding (i.e., no legal translation to a control or data value possible for 8b/10b encoding or invalid Sync Header or invalid Ordered Set for 128b/130b encoding).

- Next state of the loopback slave is Loopback.Exit if one of the following conditions apply:
 - If directed or if four consecutive EIOSs are received on any Lane. It must be noted that in 8b/10b encoding, the receiving four consecutive EIOS indicates that the Lane received four consecutive sets of COM, IDL, IDL, IDL or alternatively, two out of three K28.3 (IDL) Symbols in each of the four consecutive sets of transmitted EIOS. In 128b/130b encoding, receiving four consecutive EIOS indicates receiving the full 130-bit EIOS in the first three and the first four Symbols following a 01b Sync Header in the last EIOS.
 - Optionally, if current Link speed is 2.5 GT/s and an EIOS is received or Electrical Idle is detected/inferred on any Lane.
 - Note: As described in Section 4.2.4.3, an Electrical Idle condition may be inferred if any of the configured Lanes remained electrically idle continuously for 128 µs by not detecting an exit from Electrical Idle in the entire 128 µs window.
 - A loopback slave must be able to detect an Electrical Idle condition on any Lane within 1 ms of the EIOS being received by the loopback slave.
 - Note: During the time after an EIOS is received and before Electrical Idle is actually detected by the Loopback Slave, the Loopback Slave may receive a bit stream undefined by the encoding scheme, which may be looped back by the transmitter.
 - The $T_{TX\text{-}IDLE\text{-}SET\text{-}TO\text{-}IDLE}$ parameter does not apply in this case since the loopback slave may not even detect Electrical Idle until as much as 1 ms after the EIOS.
- The next state of the loopback master is Loopback.Exit if directed.

4.2.6.10.3. Loopback.Exit

- The loopback master sends an EIOS for Ports that support only the 2.5 GT/s data rate and eight consecutive EIOSs for Ports that support greater than 2.5 GT/s data rate, and optionally for Ports that only support the 2.5 GT/s data rate, irrespective of the current Link speed, and enters Electrical Idle on all Lanes for 2 ms.

- The loopback master must transition to a valid Electrical Idle condition[59] on all Lanes within $T_{TX-IDLE-SET-TO-IDLE}$ after sending the last EIOS.
- Note: The EIOS can be useful in signifying the end of transmit and compare operations that occurred by the loopback master. Any data received by the loopback master after any EIOS is received should be ignored since it is undefined.

- The loopback slave must enter Electrical Idle on all Lanes for 2 ms.
 - Before entering Electrical Idle the loopback slave must Loopback all Symbols that were received prior to detecting Electrical Idle. This ensures that the loopback master may see the EIOS to signify the logical end of any Loopback send and compare operations.
- The next state of the loopback master and loopback slave is Detect.

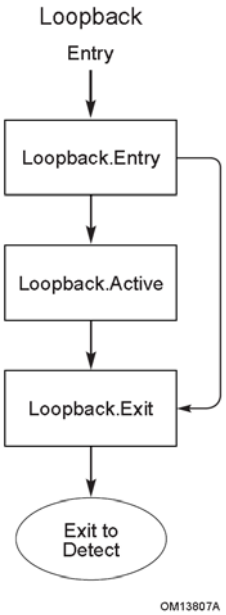


Figure 4-30: Loopback Substate Machine

4.2.6.11. Hot Reset

- Lanes that were directed by a higher Layer to initiate Hot Reset:
 - All Lanes in the configured Link transmit TS1 Ordered Sets with the Hot Reset bit asserted and the configured Link and Lane numbers.
 - If two consecutive TS1 Ordered Sets are received on any Lane with the Hot Reset bit asserted and configured Link and Lane numbers, then:

[59] The common mode being driven does not need to meet the Absolute Delta Between DC Common Mode During L0 and Electrical Idle ($V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$) specification (see Table 4-18).

- LinkUp = 0b (False)
- If no higher Layer is directing the Physical Layer to remain in Hot Reset, the next state is Detect
- Otherwise, all Lanes in the configured Link continue to transmit TS1 Ordered Sets with the Hot Reset bit asserted and the configured Link and Lane numbers.

- Otherwise, after a 2 ms timeout next state is Detect.

- Lanes that were not directed by a higher Layer to initiate Hot Reset (i.e., received two consecutive TS1 Ordered Sets with the Hot Reset bit asserted on any configured Lanes):
 - LinkUp = 0b (False)
 - If any Lane of an Upstream Port of a Switch receives two consecutive TS1 Ordered Sets with the Hot Reset bit asserted, all configured Downstream Ports must transition to Hot Reset as soon as possible.
 - Any optional crosslinks on the Switch are an exception to this rule and the behavior is system specific.
 - All Lanes in the configured Link transmit TS1 Ordered Sets with the Hot Reset bit asserted and the configured Link and Lane numbers.
 - If two consecutive TS1 Ordered Sets were received with the Hot Reset bit asserted and the configured Link and Lane numbers, the state continues to be Hot Reset and the 2 ms timer is reset.
 - Otherwise, the next state is Detect after a 2 ms timeout.

Note: Generally, Lanes of a Downstream or optional crosslink Port will be directed to Hot Reset, and Lanes of an Upstream or optional crosslink Port will enter Hot Reset by receiving two consecutive TS1 Ordered Sets with the Hot Reset bit asserted on any configured Lanes, from Recovery.Idle state.

4.2.7. Clock Tolerance Compensation

SKP Ordered Sets (defined below) are used to compensate for differences in frequencies between bit rates at two ends of a Link. The Receiver Physical Layer logical sub-block must include elastic buffering which performs this compensation. The interval between SKP Ordered Set transmissions is derived from the absolute value of the Transmit and Receive clock frequency difference specified in Table 4-18. ~~Having worse case clock frequencies at the limits of the tolerance specified will result in a 600 ppm difference between the Transmit and Receive clocks of a Link. As a result, on average, the Transmit and Receive clocks can shift one clock every 1666 clocks.~~

The specification supports two kinds of clocking where the Tx and Rx Refclk rates differ. One allows for a worst case 600 ppm difference with no SSC (Separate Reference Clocks With No SSC - SRNS), and the other for a 5600 ppm difference for separate refclks utilizing independent SSC (Separate Reference Clocks with Independent SSC - SRIS) (SSC introduces a 5000 ppm difference, and Tx/Rx crystal tolerance introduces another 600 ppm). Note that the common Refclk architecture utilizes the same Refclk for Tx and Rx and so does not introduce any difference between the Tx and Rx Refclk rates.

Specific form factor specifications are permitted to require the use of only SRIS, only SRNS, or to provide a mechanism for clocking architecture selection. Upstream Ports are permitted to implement support for any combination of SRIS and SRNS (including no support for either), but must conform to the requirements of any associated form factor specification. Downstream Ports supporting SRIS must also support SRNS unless the port is only associated with a specific form factor(s) which modifies these requirements. Port configuration to satisfy the requirements of a specific associated form factor is implementation specific.

If the clock tolerance is 600 ppm, on average, the Tx and Rx clocks can shift one clock every 1666 clocks. If the clock tolerance is 5600 ppm, a shift of one clock can occur every 178 clocks.

If the receiver is capable of operating with SKP Ordered Sets being generated at the rate used in SRNS even though the Port is running in SRIS, the Port is permitted to Set its bit for the appropriate data rate in the Lower SKP OS Reception Supported Speeds Vector field of the Link Capabilities 2 register. If the transmitter is capable of operating with SKP Ordered Sets being generated at the rate used in SRNS even though the Port is running in SRIS, the Port is permitted to Set its bit for the appropriate data rate in the Lower SKP OS Generation Supported Speeds Vector field of the Link Capabilities 2 register. System software must check that the bit is Set in the Lower SKP OS Reception Supported Speeds Vector field before setting the appropriate data rate's bit in the link partner's Enable Lower SKP OS Generation Vector field of the Link Control 3 register. Any software transparent extension devices (such as a repeater) present on a Link must also support lower SKP OS Generation for system software to set the bit in the Enable Lower SKP OS Generation Vector field. Software determination of such support in such extension devices is implementation specific. When the bit for the data rate that the link is running in is Set in the Enable Lower SKP OS Generation Vector field, the transmitter schedules SKP Ordered Set generation in L0 at the rate used in SRNS, regardless of which clocking architecture the link is running in. In other LTSSM states, SKP Ordered Set scheduling is at the appropriate rate for the clocking architecture.

Components supporting SRIS may need more entries in their elastic buffers than designs supporting SRNS only. This requirement takes into account the extra time it may take to schedule a SKP Ordered Set if the latter falls immediately after a maximum payload sized packet.

4.2.7.1. SKP Ordered Set for 8b/10b Encoding

When using 8b/10b encoding, a transmitted SKP Ordered Set is a COM Symbol followed by three SKP Symbols, except as is allowed for a Loopback Slave in the Loopback.Active LTSSM state. A received SKP Ordered Set is a COM Symbol followed by one to five SKP Symbols.

4.2.7.2. SKP Ordered Set for 128b/130b Encoding

Table 4-15 describes the layout of the SKP Ordered Set when using 128b/130b encoding. A transmitted SKP Ordered Set is 16 Symbols. Four SKP Symbols can be added and removed by a Port. Hence, a received SKP Ordered Set can be 8, 12, 16, 20, or 24 Symbols. The SKP_END Symbol indicates the last four Symbols of SKP Ordered Set so that receivers can identify the location of the next Block Sync Header in the bit stream. The three Symbols following the SKP_END Symbol contain different information depending on the LTSSM state.

The Data Parity bit in Table 4-15 is the even parity of the payload of all the Data Blocks transmitted after scrambling since the last SDS Ordered Set or SKP Ordered Set, whichever Ordered Set was transmitted most recently. The Data Parity bit is calculated for each Lane independently. Receivers are required to calculate the parity using the same method as Transmitters and compare the calculated result with the received Data Parity bit. If the calculated and received values do not match, the bit in the Lane Error Status register, corresponding to the default Lane number where the mismatch was detected, must be set. However, such a mismatch is not a Receiver Error and must not cause a Link retrain.

The Error_Status field, defined for Modified Compliance Pattern, contains two pieces of information: Pattern Lock (Error_Status[7]) and Receiver Error Count (Error_Status[6:0]). See Section 4.2.6.2.2 for more information.



IMPLEMENTATION NOTE

LFSR in SKP Ordered Set

The LFSR value is transmitted to enable trace tools to be able to function even if they need to reacquire block alignment in the midst of a bit stream. Since trace tools cannot force the link to enter Recovery, they can reacquire bit lock, if needed, and monitor for the SKP Ordered Set to obtain Block alignment and perform Lane-to-Lane de-skew. The LFSR value from the SKP Ordered Set can be loaded into its LFSR to start interpreting the bit stream. It must be noted that with a bit stream one may alias to a SKP Ordered Set on a non-Block boundary. The trace tools can validate their Block alignment by using implementation specific means such as receiving a fixed number of valid packets or Sync Headers or subsequent SKP Ordered Sets.

Table 4-15: SKP Ordered Set with 128b/130b Encoding

Symbol Number	Value	Description
0 through (4*N - 1) [N can be 1 through 5]	AAh	SKP Symbol. Symbol 0 is the SKP Ordered Set identifier.
4*N	E1h	SKP_END Symbol. Signifies the end of the SKP Ordered Set after three more Symbols.
4*N + 1	00-FFh	(i) If LTSSM state is Polling.Compliance: AAh (ii) Else if prior block was a Data Block: Bit[7] = Data Parity Bit[6:0] = LFSR[22:16] (iii) Else: Bit[7] = ~LFSR[22] Bit[6:0] = LFSR[22:16]
4*N + 2	00-FFh	(i) If the LTSSM state is Polling.Compliance: Error_Status[7:0] (ii) Else LFSR[15:8]
4*N + 3	00-FFh	(i) If the LTSSM state is Polling.Compliance: ~Error_Status[7:0] (ii) Else: LFSR[7:0]

4.2.7.3. Rules for Transmitters

- All Lanes shall transmit Symbols at the same frequency (the difference between bit rates is 0 ppm within all multi-Lane Links).
- When transmitted, SKP Ordered Sets of the same length shall be transmitted simultaneously on all Lanes of a multi-Lane Link, except as allowed for a Loopback Slave in the Loopback.Active LTSSM State (see Section 4.2.4.10 and Table 4-18 for the definition of simultaneous in this context).
- The transmitted SKP Ordered Set when using 8b/10b encoding must follow the definition in Section 4.2.7.1.
- The transmitted SKP Ordered Set when using 128b/130b encoding must follow the definition in Section 4.2.7.2.
- When using 8b/10b encoding: If the Link is not operating in ~~SRNS, or in~~ SRIS, or ~~and~~the bit corresponding to the current Link speed is Set in the Enable Lower SKP OS Generation Vector field and the LTSSM is in L0, a ~~A~~ SKP Ordered Set must be scheduled for transmission at an interval between 1180 and 1538 Symbol Times. If the Link is operating in SRIS and either the bit corresponding to the current Link speed is Clear in the Enable Lower SKP OS Generation Vector field or the LTSSM is not in L0, a SKP Ordered Set must be scheduled for transmission at an interval of less than 154 Symbol Times.
- ~~-~~When using 128b/130b encoding: If the Link is not operating ~~in SRNS or~~ in SRIS, or ~~and~~the bit corresponding to the current Link speed is Set in the Enable Lower SKP OS Generation Vector field and the LTSSM is in L0, a ~~A~~ SKP Ordered Set must be scheduled for transmission at an interval between 370 to 375 blocks, when the LTSSM is not in the Polling.Compliance state and is not in the Loopback state, ~~or~~ unless the Transmitter is a Loopback Slave that has not started looping back the incoming bit stream. If the Link is operating in SRIS and either the bit corresponding to the current Link speed is Clear in the Enable Lower SKP OS Generation Vector field or the LTSSM is not in L0, a SKP Ordered Set must be scheduled for transmission at an interval less than 38 blocks, when the LTSSM is not in the Loopback state or is a Loopback Slave that has not started looping back the incoming bit stream. When the LTSSM is in the Loopback state and the Link is operating in SRNS, the Loopback Master must schedule two SKP Ordered Sets to be transmitted, at most two blocks apart from each other, at an interval between 370 to 375 blocks. When the LTSSM is in the Loopback state and the Link is operating in SRIS, the Loopback Master must schedule two SKP Ordered Sets to be transmitted, at most two blocks apart from each other, at an interval of less than 38 blocks.
- Scheduled SKP Ordered Sets shall be transmitted if a packet or Ordered Set is not already in progress, otherwise they are accumulated and then inserted consecutively at the next packet or Ordered Set boundary. Note: When using 128b/130b encoding, SKP Ordered Sets cannot be transmitted in consecutive Blocks within a Data Stream. See Section 4.2.2.3.2 for more information.
- SKP Ordered Sets do not count as an interruption when monitoring for consecutive Symbols or Ordered Sets (e.g., eight consecutive TS1 Ordered Sets in Polling.Active).

Comment [VS139]: Errata B34 (Part C list resolved 10/13)

Comment [VS140]: ECN: SRIS

Comment [VS141]: Errata B34 (Part C list resolved 10/13)

Comment [VS142]: ECN:SRIS

Comment [VS143]: Errata A25 (Part C list resolved 10/13)

Comment [VS144]: ECN: SRIS

- When using 8b/10b encoding: SKP Ordered Sets must not be transmitted while the Compliance Pattern or the Modified Compliance Pattern (see Section 4.2.8) is in progress during Polling.Compliance if the Compliance SOS bit of the Link Control 2 register is 0b. If the Compliance SOS bit of the Link Control 2 register is 1b, two consecutive SKP Ordered Sets must be sent (instead of one) for every scheduled SKP Ordered Set time interval while the Compliance Pattern or the Modified Compliance Pattern is in progress when using 8b/10b encoding.
- When using 128b/130b encoding: The Compliance SOS register bit has no effect. While in Polling.Compliance, Transmitters must not transmit any SKP Ordered Sets other than those specified as part of the Modified Compliance Pattern in Section 4.2.11.
- Any and all time spent in a state when the Transmitter is electrically idle does not count in the scheduling interval used to schedule the transmission of SKP Ordered Sets.
- It is recommended that any counter(s) or other mechanisms used to schedule SKP Ordered Sets be reset any time when the Transmitter is electrically idle.

4.2.7.4. Rules for Receivers

- Receivers shall recognize received SKP Ordered Sets as defined in Section 4.2.7.1 when using 8b/10b encoding and as defined in Section 4.2.7.2 when using 128b/130b encoding.
 - The length of the received SKP Ordered Sets shall not vary from Lane-to-Lane in a multi-Lane Link, except as may occur during Loopback.Active.
- Receivers shall be tolerant to receive and process SKP Ordered Sets at an average interval between 1180 to 1538 Symbol Times when using 8b/10b encoding and 370 to 375 blocks when using 128b/130b encoding when the Link is not operating in ~~SRNS or in~~ SRIS or ~~and~~ its bit for the current Link speed is Set in the Lower SKP OS Reception Supported Speeds Vector field. Receivers shall be tolerant to receive and process SKP Ordered Sets at an average interval of less than 154 Symbol Times when using 8b/10b encoding and less than 38 blocks when using 128b/130b encoding when the Link is operating in SRIS.
 - Note: Since Transmitters in electrical idle are not required to reset their mechanism for time-based scheduling of SKP Ordered Sets, Receivers shall be tolerant to receive the first time-scheduled SKP Ordered Set following electrical idle in less than the average time interval between SKP Ordered Sets.
- For 8.0 GT/s and above data rates, in L0 state, Receivers must check that each SKP Ordered Set is preceded by a Data Block with an EDS packet.
- Receivers shall be tolerant to receive and process consecutive SKP Ordered Sets in 2.5 GT/s and 5.0 GT/s data rates.
 - Receivers shall be tolerant to receive and process SKP Ordered Sets that have a maximum separation dependent on the Max_Payload_Size a component supports. For 2.5 GT/s and 5.0 GT/s data rates, the formula for the maximum number of Symbols (N) between SKP Ordered Sets is: N = 1538 + (Max_payload_size_byte+28). For example, if Max_Payload_Size is 4096 bytes, N = 1538 + 4096 + 28 = 5662.

Comment [VS145]: Errata A25 (Part C list resolved 10/13)

Comment [VS146]: Errata B34 (Part C list resolved 10/13)

Comment [VS147]: ECN: SRIS

4.2.8. Compliance Pattern in 8b/10b Encoding

During Polling, the Polling.Compliance substate must be entered from Polling.Active based on the conditions described in Section 4.2.6.2.1. The compliance pattern consists of the sequence of 8b/10b Symbols K28.5, D21.5, K28.5, and D10.2 repeating. The Compliance sequence is as follows:

Symbol	K28.5	D21.5	K28.5	D10.2
Current Disparity	Negative	Positive	Positive	Negative
Pattern	0011111010	1010101010	1100000101	0101010101

The first Compliance sequence Symbol must have negative disparity. It is permitted to create a disparity error to align the running disparity to the negative disparity of the first Compliance sequence Symbol.

For any given device that has multiple Lanes, every eighth Lane is delayed by a total of four Symbols. A two Symbol delay occurs at both the beginning and end of the four Symbol Compliance Pattern sequence. A x1 device, or a xN device operating a Link in x1 mode, is permitted to include the Delay Symbols with the Compliance Pattern.

This delay sequence on every eighth Lane is then:

Symbol:	D	D	K28.5	D21.5	K28.5	D10.2	D	D

Where D is a K28.5 Symbol. The first D Symbol has negative disparity to align the delay disparity with the disparity of the Compliance sequence.

After the eight Symbols are sent, the delay Symbols are advanced to the next Lane, until the delay Symbols have been sent on all eight lanes. Then the delay Symbols cycle back to Lane 0, and the process is repeated. It is permitted to advance the delay sequence across all eight lanes, regardless of the number of lanes detected or supported. An illustration of this process is shown below:

Lane 0	D	D	K28.5-	D21.5	K28.5+	D10.2	D	D	K28.5-	D21.5	K28.5+	D10.2
Lane 1	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	D	D	K28.5-	D21.5
Lane 2	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2
Lane 3	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2
Lane 4	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2
Lane 5	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2
Lane 6	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2
Lane 7	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2
Lane 8	D	D	K28.5-	D21.5	K28.5+	D10.2	D	D	K28.5-	D21.5	K28.5+	D10.2
Lane 9	K28.5-	D21.5	K28.5+	D10.2	K28.5-	D21.5	K28.5+	D10.2	D	D	K28.5-	D21.5

Key:

K28.5-	COM when disparity is negative, specifically: “0011111010”
K28.5+	COM when disparity is positive, specifically: “1100000101”
D21.5	Out of phase data Symbol, specifically: “1010101010”
D10.2	Out of phase data Symbol, specifically: “0101010101”
D	Delay Symbol K28.5 (with appropriate disparity)

This sequence of delays ensures interference between adjacent Lanes, enabling measurement of the compliance pattern under close to worst-case Inter-Symbol Interference and cross-talk conditions.

4.2.9. Modified Compliance Pattern in 8b/10b Encoding

The Modified Compliance Pattern consists of the same basic Compliance Pattern sequence (see Section 4.2.8) with one change. Two identical error status Symbols followed by two K28.5 are appended to the basic Compliance sequence of 8b/10b Symbols (K28.5, D21.5, K28.5, and D10.2) to form the Modified Compliance Sequence of (K28.5, D21.5, K28.5, D10.2, error status Symbol, error status Symbol, K28.5, K28.5). The first Modified Compliance Sequence Symbol must have negative disparity. It is permitted to create a disparity error to align the running disparity to the negative disparity of the first Modified Compliance Sequence Symbol. For any given device that has multiple Lanes, every eighth Lane is moved by a total of eight Symbols. Four Symbols of K28.5 occurs at the beginning and another four Symbols of K28.7 occurs at the end of the eight Symbol Modified Compliance Pattern sequence. The first D Symbol has negative disparity to align the delay disparity with the disparity of the Modified Compliance Sequence. After the 16 Symbols are sent, the delay Symbols are advanced to the next Lane, until the delay Symbols have been sent on all eight lanes. Then the delay Symbols cycle back to Lane 0, and the process is repeated. It is permitted to advance the delay sequence across all eight lanes, regardless of the number of lanes detected or supported. A x1 device, or a xN device operating a Link in x1 mode, is permitted to include the Delay symbols with the Modified Compliance Pattern.

An illustration of the Modified Compliance Pattern is shown below:

Lane0	D	D	D	D	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.7-	K28.7-	K28.7-	K28.7-	K28.5-	D21.5
Lane1	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	D	D
Lane2	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5
Lane3	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5
Lane4	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5
Lane5	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5
Lane6	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5
Lane7	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5
Lane8	D	D	D	D	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.7-	K28.7-	K28.7-	K28.7-	K28.5-	D21.5
Lane9	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	K28.5-	D21.5	K28.5+	D10.2	ERR	ERR	K28.5-	K28.5+	D	D

Key:

K28.5-	COM when disparity is negative, specifically: “0011111010”
K28.5+	COM when disparity is positive, specifically: “1100000101”
D21.5	Out of phase data Symbol specifically: “1010101010”
D10.2	Out of phase data Symbol, specifically: “0101010101”
D	Delay Symbol K28.5 (with appropriate disparity)
ERR	error status Symbol (with appropriate disparity)
K28.7-	EIE when disparity is negative, specifically “0011111000”

The reason two identical error Symbols are inserted instead of one is to ensure disparity of the 8b/10b sequence is not impacted by the addition of the error status Symbol.

All other Compliance pattern rules are identical (i.e., the rules for adding delay Symbols) so as to preserve all the crosstalk characteristics of the Compliance Pattern.

The error status Symbol is an 8b/10b data Symbol, maintained on a per Lane basis, and defined in 8-bit domain in the following way:

- Receiver Error Count (Bits 6:0) - Incremented on every Receiver error after the Pattern Lock bit becomes asserted.
- Pattern Lock (Bit 7) - Asserted when the Lane locks to the incoming Modified Compliance Pattern.

Formatted: bullet, Indent: Left: 0", Right: 0", Tab stops: Not at 0.5"

4.2.10. Compliance Pattern in 128b/130b Encoding

The compliance pattern consists of the following repeating sequence of 36 Blocks

1. One block with a Sync Header of 01b followed by a 128-bit unscrambled payload of 64 1's followed by 64 0's
2. One block with a Sync Header of 01b followed by a 128-bit unscrambled payload of the following:

	Lane No modulo 8 = 0	**Lane No modulo 8 = 1**	**Lane No modulo 8 = 2**	**Lane No modulo 8 = 3**	**Lane No modulo 8 = 4**	**Lane No modulo 8 = 5**	**Lane No modulo 8 = 6**	**Lane No modulo 8 = 7**
Symbol 0	55h	FFh	FFh	FFh	55h	FFh	FFh	FFh
Symbol 1	55h	FFh	FFh	FFh	55h	FFh	FFh	FFh
Symbol 2	55h	00h	FFh	FFh	55h	FFh	FFh	FFh
Symbol 3	55h	00h	FFh	FFh	55h	FFh	F0h	F0h
Symbol 4	55h	00h	FFh	C0h	55h	FFh	00h	00h
Symbol 5	55h	00h	C0h	00h	55h	E0h	00h	00h
Symbol 6	55h	00h	00h	00h	55h	00h	00h	00h
Symbol 7	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}
Symbol 8	00h	1Eh	2Dh	3Ch	4Bh	5Ah	69h	78h
Symbol 9	00h	55h	00h	00h	00h	55h	00h	F0h
Symbol 10	00h	55h	00h	00h	00h	55h	00h	00h
Symbol 11	00h	55h	00h	00h	00h	55h	00h	00h
Symbol 12	00h	55h	0Fh	0Fh	00h	55h	07h	00h
Symbol 13	00h	55h	FFh	FFh	00h	55h	FFh	00h
Symbol 14	00h	55h	FFh	FFh	7Fh	55h	FFh	00h
Symbol 15	00h	55h	FFh	FFh	FFh	55h	FFh	00h

Key: P: Indicates the 4-bit encoding of the Transmitter preset being used.
~P: Indicates the bit-wise inverse of P.

3. One block with a Sync Header of 01b followed by a 128-bit unscrambled payload of the following:

	Lane No modulo 8 = 0	Lane No modulo 8 = 1	Lane No modulo 8 = 2	Lane No modulo 8 = 3	Lane No modulo 8 = 4	Lane No modulo 8 = 5	Lane No modulo 8 = 6	Lane No modulo 8 = 7
Symbol 0	FFh	FFh	55h	FFh	FFh	FFh	55h	FFh
Symbol 1	FFh	FFh	55h	FFh	FFh	FFh	55h	FFh
Symbol 2	FFh	FFh	55h	FFh	FFh	FFh	55h	FFh
Symbol 3	F0h	F0h	55h	F0h	F0h	F0h	55h	F0h
Symbol 4	00h	00h	55h	00h	00h	00h	55h	00h
Symbol 5	00h	00h	55h	00h	00h	00h	55h	00h
Symbol 6	00h	00h	55h	00h	00h	00h	55h	00h
Symbol 7	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}	{P,~P}
Symbol 8	00h	1Eh	2Dh	3Ch	4Bh	5Ah	69h	78h
Symbol 9	00h	00h	00h	55h	00h	00h	00h	55h
Symbol 10	00h	00h	00h	55h	00h	00h	00h	55h
Symbol 11	00h	00h	00h	55h	00h	00h	00h	55h
Symbol 12	FFh	0Fh	0Fh	55h	0Fh	0Fh	0Fh	55h
Symbol 13	FFh	FFh	FFh	55h	FFh	FFh	FFh	55h
Symbol 14	FFh	FFh	FFh	55h	FFh	FFh	FFh	55h
Symbol 15	FFh	FFh	FFh	55h	FFh	FFh	FFh	55h

Key: P: Indicates the 4-bit encoding of the Transmitter preset being used.
~P: Indicates the bit-wise inverse of P.

4. One EIEOS Block
5. 32 Data Blocks, each with a payload of 16 IDL data Symbols (00h) scrambled



IMPLEMENTATION NOTE

First Two Blocks of the Compliance Pattern

The first block is a very low frequency pattern to help with measurement of the preset settings. The second block is to notify the Lane number and preset encoding the compliance pattern is using along with ensuring the entire compliance pattern is DC Balanced.

The payload in each Data Block is the output of the scrambler in that Lane (i.e., input data is 0b). The scrambler does not advance during the Sync Header bits. The scrambler is initialized when an EIEOS is transmitted. The Lane numbers used to determine the scrambling LFSR seed value depend on how Polling.Compliance is entered. If it is entered due to the Enter Compliance bit in the Link Control 2 register being set, then the Lane numbers are the numbers that were assigned to the Lanes and the Receiver Lane polarity to be used on each Lane is the Lane polarity inversion that was used in the most recent time that LinkUp was 1b. If a Lane was not part of the configured Link

at that time, and for all other methods of entering Polling.Compliance, the Lane numbers are the default numbers assigned by the Port. These default numbers must be unique. For example, each Lane of a x16 Link must be assigned an unique Lane number between 0 to 15. The Data Blocks of the compliance pattern do not form a Data Stream and hence are exempt from the requirement of transmitting an SDS Ordered Set or EDS Token during Ordered Set Block to Data Block transition and vice-versa.

4.2.11. Modified Compliance Pattern in 128b/130b Encoding

The modified compliance pattern, when not operating in SRIS, consists of repeating the following sequence of 65792 Blocks:

1. One EIEOS Block
2. 256 Data Blocks, each with a payload of 16 Idle data Symbols (00h), scrambled
3. 255 sets of the following sequence:
 i. One SKP Ordered Set
 ii. 256 Data Blocks, each with a payload of 16 Idle data Symbols (00h), scrambled

Comment [VS148]: Errata A26 (Part C list resolved 10/13)

The modified compliance pattern, when operating in SRIS, consists of repeating the following sequence of 67585 Blocks:

1. One EIEOS Block
2. 2048 sets of the following sequence:
 i. One SKP Ordered Set
 ii. 32 Data Blocks, each with a payload of 16 Idle data Symbols (00h), scrambled

Comment [VS149]: Errata A26 (Part C list resolved 10/13)

The payload in each Data Block is the output of the scrambler in that Lane (i.e., input data is 0b). The scrambler does not advance during the Sync Header bits. The scrambler is initialized when an EIEOS is transmitted. The Lane numbers used to determine the scrambling LFSR seed value depend on how Polling.Compliance is entered. If it is entered due to the Enter Compliance bit in the Link Control 2 register being set, then the Lane numbers are the numbers that were assigned to the Lanes and the Receiver Lane polarity to be used on each Lane is the Lane polarity inversion used in the most recent time that LinkUp was 1b. If a Lane was not part of the configured Link at that time, and for all other methods of entering Polling.Compliance, the Lane numbers are the default numbers assigned by the Port. These default numbers must be unique. For example, each Lane of a x16 Link must be assigned an unique Lane number from 0 to 15. The Data Blocks of the modified compliance pattern do not form a Data Stream and hence are exempt from the requirement of transmitting an SDS Ordered Set or EDS Token during Ordered Set Block to Data Block transition and vice-versa.

4.3. Electrical Sub-block

4.3.1. Electrical Specification Organization

The 3.0 version of the PCI Express electrical specification is organized into separate sections for the transmitter, receiver, the channel, and the Refclk. Each of these sections is further partitioned according to data rate: 2.5/5.0, or 8.0 GT/s. As the data rate has evolved upward it has been necessary to modify what parameters are defined to guarantee interoperability as well as how those parameters are measured. For 8.0 GT/s, in particular, most high frequency parameters need to be measured indirectly and then computed via postprocessing.

Due to the large differences in the parameter definition and measurement between 2.5/5.0 and 8.0 GT/s the text describing the parameters is located in separate subsections. Where parameters are common among all three data rates, they appear in a single table with columns for each of the three data rates.

4.3.2. Interoperability Criteria for 2.5, 5.0, and 8.0 GT/s Devices

The PCI Express electrical specification is constructed to guarantee backwards compatibility among devices of all three speeds. Since all devices power up at 2.5 GT/s interoperability at that speed is guaranteed. Operation at 5.0 or 8.0 GT/s requires that the two devices negotiate to a higher data rate. Details of this process are defined in the protocol section.

8.0 GT/s ~~root Port~~ devices must support 2.5, 5.0, and 8.0 GT/s operation~~, because a downstream device could support any of these data rates. A downstream device must support 2.5 GT/s and may, support either 5.0 or 8.0 GT/s operation, or both~~. 5.0 GT/s devices must support 2.5 and 5.0 GT/s operation. See also Table 4-5 and Table 4-6.

Comment [VS150]: Errata A1

The Refclk is defined as part of the 3.0 electrical specification with the expectation that the characteristics which define a minimally compliant Refclk are approximately the same for all three data rates. In other words, operation at the higher data rates is not predicated upon improving the jitter characteristics of the Refclk.

Data rates of 5.0 GT/s and 8.0 GT/s specify a procedure for tolerancing the channel, where the characteristics of a minimally compliant channel for 5.0 and 8.0 GT/s are approximately the same.

4.3.2.1. 8.0 GT/s Standards Applied to 2.5 and 5.0 GT/s Devices

Evolving process technology and circuit characteristics which enable 8.0 GT/s performance imposes limitations on a small number of parameters defined for 2.5 and 5.0 GT/s operation. The merged 2.5/5.0/8.0 GT/s specification addresses these limitations by relaxing or extending certain 2.5 and 5.0 GT/s parameters. These changes are also reflected in the relevant parameter tables.

4.3.2.1.1. ESD Standards

All PCI Express signal and power supply pins must be tested for ESD protection levels to the Human Body Model (HBM) and the Charged Device Model (CDM) standards in accordance with ESDA/JEDEC JS-001-2010 (for HBM) and in accordance with JEDEC JESD22-C101 (for CDM). Pins must meet or exceed the minimum levels recommended in JEDEC JEP155/JEP157 (HBM/CDM) or JEDEC approved superseding documents. Note that this section is applicable for 2.5, 5.0, and 8.0 GT/s capable devices.

4.3.2.1.2. Receiver Powered Off Resistance

The $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}POS}$ and $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}NEG}$ parameter values defined for 8.0 GT/s will be applied to 2.5 and 5.0 GT/s devices.

4.3.2.1.3. Tx Equalization Tolerance

The tolerance for TxEQ de-emphasis for 2.5 and 5.0 GT/s signaling is relaxed from ±0.5 dB to ±1.0 dB. This change makes the -3.5 and -6.0 dB de-emphasis tolerance consistent for all three data rates.

4.3.2.1.4. Tx Equalization During Tx Margining

The 2.5 and 5.0 GT/s specifications state that the ±0.5 dB TxEQ tolerance is relaxed to ±1.0 dB during Tx margining. For 8.0 GT/s, the accuracy that can be achieved for TxEQ settings is determined by the Tx coefficient granularity and the TxEQ tolerances specified for the transmitter during normal operation.

4.3.2.1.5. $V_{TX\text{-}ACCM}$ and $V_{RX\text{-}ACCM}$

For 8.0 GT/s both the Tx ACCM and the Rx ACCM limits have changed with respect to those defined for 2.5 GT/s and 5.0 GT/s. For both 2.5 and 5.0 GT/s, $V_{TX\text{-}ACCM}$ is relaxed to 150 mVPP, but a separate LF component is not defined.

Rx AC CM limit of 300 mVPP for 2.5/5.0 GT/s remains unchanged. The 8.0 GT/s AC CM limits are defined in Section 4.3.4.4.1 and Table 4-22.

4.3.2.2. 2.5 GT/s is Not a Subset of 5.0 GT/s and 5.0 GT/s is Not a Subset of 8.0 GT/s

Since a 5.0 GT/s device must be capable of operation at either 2.5 GT/s or 5.0 GT/s data rates, the device must meet the 2.5 GT/s and 5.0 GT/s specifications in their entirety. Meeting only the 5.0 GT/s specification does not guarantee interoperability at 2.5 GT/s speeds, as is described in the following example.

The 5.0 GT/s clock specification requires a tighter window than 2.5 GT/s (8-16 MHz vs. 1.5-22 MHz). Imagine a clock source that just meets the 5.0 GT/s Refclk jitter specification, but has a huge jitter spike at 3 MHz (or 20 MHz). This spike would not be caught by the 8-16 MHz range,

but could potentially fail the 2.5 GT/s component. Therefore, the clocks need to pass both the 2.5 GT/s and 5.0 GT/s limits.

4.3.2.3. Component Interfaces

PCI Express components from different manufacturers must interoperate reliably together. At the electrical level, this is achieved by specifying a set of parameters and the interfaces at which those parameters must be met. For 5.0 GT/s PCI Express components, this interface is defined to be at the pins of the Receiver and Transmitter devices and the corresponding locations at either end of the channel. At 2.5 GT/s the requirement to reference all measurements to the Tx or Rx pins is less explicitly stated in the specification, but still holds true. The pin location was also chosen because it permits a component to be characterized independently of others. For example, a Transmitter can be measured without needing any other PCI Express components.

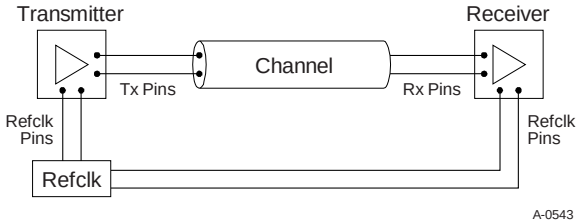


Figure 4-31: Transmitter, Channel, and Receiver Boundaries

Additional interfaces may be specified at the connector or connectors for multi-segment topologies, such as between server blades and backplane or between a motherboard and adapter. Due to the large number of possible combinations and their product specific nature, multi-segment specifications will be detailed in the *PCI Express Card Electromechanical Specification* and other form factor specifications.

4.3.3. Transmitter Specification

4.3.3.1. Transmitter Pin to Pad Correlation

The PCI Express Specification defines Transmitter and Receiver measurements at the pin, while a pad location is preferred by silicon designers. Measurements to specification are referenced to a device's pins, because this is the only location that is generally accessible for probing (die-pads in an assembled device are not usually accessible). It is the shared responsibility of the silicon and the package designers to account for silicon package interactions and guarantee positive margins at the device's pins.

4.3.3.1.1. Measurement Setup for Characterizing 2.5 GT/s and 5.0 GT/s Transmitters

A Transmitter's parameters may be characterized by means of the test setup shown in Figure 4-32 and Figure 4-33. When measuring a Transmitter, it is not usually feasible to place the probes directly at the Transmitter's pins, so it is typical to have PCB traces and other structures between the Tx

package and the probe location. If direct measurement cannot be made at the Tx pins, then it will be necessary to deconvolve the effects compliance test board from the measurement. Coupling capacitors are included, since some Transmitters will need them to establish the correct DC bias. If a Transmitter can operate with a 0.0 V DC bias or if the test equipment can provide the correct bias, then the coupling capacitors may be eliminated. The range of acceptable values for the coupling capacitors appears in Table 4-18.

The impedance targets for the interconnects shown in Figure 4-32 and Figure 4-33 are 100 Ω differential and 50 Ω single-ended.

Measurements made at 2.5 GT/s may be achieved by locating the probe points close to the DUT and not de-embedding the test fixture, as shown in Figure 4-33. Measurements at 5.0 GT/s must de-embed the test fixture, as shown in Figure 4-32. It is also acceptable to use a common test fixture and de-embed it for measurements at both 2.5 and 5.0 GT/s.

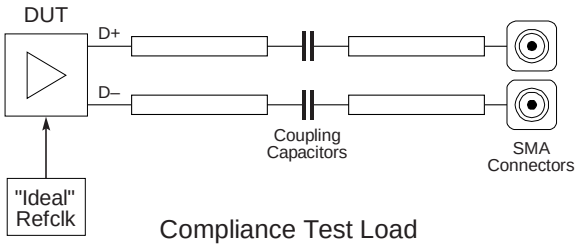


Figure 4-32: Required Setup for Characterizing a 5.0 GT/s Transmitter

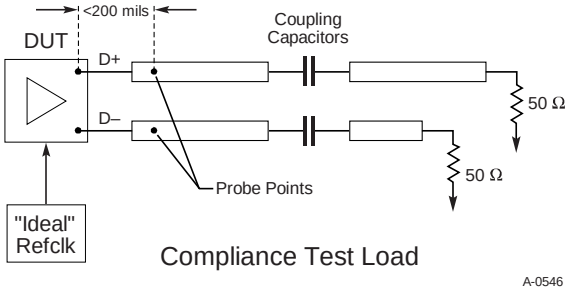


Figure 4-33: Allowable Setup for Characterizing a 2.5 GT/s Transmitter

4.3.3.1.2. Measurement Setup for Characterizing 8.0 GT/s Transmitters

The PCI Express electrical specification references all measurements to the device's pin. However, the pin of a device under test (DUT) is not generally accessible, and the closest accessible point is usually a pair of microwave-type coaxial connectors separated from the DUT pins by several inches of PCB trace, called the breakout channel. On a test board with many lanes the minimum breakout channel length is constrained by the need to route to a large number of coaxial connectors. Typically, this limitation holds true for both the Tx and the Rx pins. Figure 4-34 illustrates a typical test connection to a DUT, showing a single Tx lane breakout.

A low jitter Refclk source is required in order that the jitter measurements for the DUT include only contributions from the latter.

When testing a transmitter it is desirable to have as many other PCI Express Lanes sending or receiving data as is feasible. Similarly, if the device supports other I/O it should also be sending or receiving on these interfaces. The goal is to have the Tx test environment replicate that found in a real system as closely as possible.

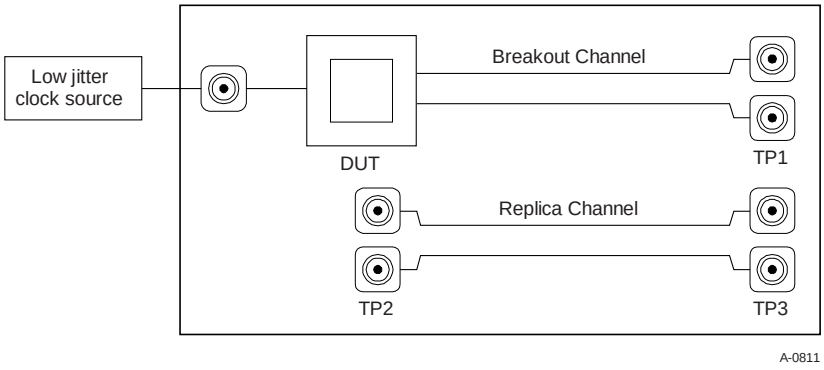


Figure 4-34: Tx Test Board Example

4.3.3.1.3. 8.0 GT/s Breakout and Replica Channels

In order to specify a transmitter with a uniform set of Tx parameters it is necessary to establish a one-to-one correspondence between what is measurable at TP1 and the corresponding Tx voltage or jitter parameter at the pin. This may be achieved by means of a breakout channel and a replica channel. The replica channel reproduces the electrical characteristics of the breakout channel as closely as possible, matching its length, layer transitions, etc, making it possible to de-embed Tx measurements to the pin of the DUT. While the specification does not define precise electrical characteristics for the replica and breakout channels, it is advisable to adhere to the following guidelines:

- Breakout channels should be as short as feasible, less than 6 inches if possible. The longer the channel the more HF information will be lost.
- Breakout channels should be the same length and routed on as few layers as possible, thereby reducing the number of replica channels that need to be built and measured.
- Each routing layer on a test board should have a separate breakout channel where the via and pad structures of the breakout and replica channels on respective layers match as closely as possible.
- Break-out channels should be designed to have an insertion loss of less than 2 dB and a return loss of greater than 15 dB at 4 GHz, which may require use of low loss dielectric, wide signal traces and back-drilling of break-out vias or use of micro-via technology.
- The impedance targets for the breakout channel are 100 Ω differential and 50 Ω single-ended. For best accuracy the actual breakout channel impedance should be within ±10% of these values. For larger deviations a more complex de-embedding technique may be required.

4.3.3.2. Voltage Level Definitions

A differential voltage is defined by taking the voltage difference between two conductors. In this specification, a differential signal or differential pair is comprised of a voltage on a positive conductor, V_{D+}, and a negative conductor, V_{D-}. The differential voltage (V_{DIFF}) is defined as the difference of the positive conductor voltage and the negative conductor voltage ($V_{DIFF} = V_{D+} - V_{D-}$). The Common Mode Voltage (V_{CM}) is defined as the average or mean voltage present on the same differential pair ($V_{CM} = [V_{D+} + V_{D-}]/2$). This document's electrical specifications often refer to peak-to-peak measurements or peak measurements, which are defined by the following equations.

- $V_{DIFFp\text{-}p} = (2*\max|V_{D+} - V_{D-}|)$ (This applies to a symmetric differential swing)
- $V_{DIFFp\text{-}p} = (\max|V_{D+} - V_{D-}|\ \{V_{D+} > V_{D-}\} + \max|V_{D+} - V_{D-}|\ \{V_{D+} < V_{D-}\})$ (This applies to an asymmetric differential swing.)
- $V_{DIFFp} = (\max|V_{D+} - V_{D-}|)$ (This applies to a symmetric differential swing.)
- $V_{DIFFp} = (\max|V_{D+} - V_{D-}|\ \{V_{D+} > V_{D-}\})$ or $(\max|V_{D+} - V_{D-}|\ \{V_{D+} < V_{D-}\})$ whichever is greater (This applies to an asymmetric differential swing.)
- $V_{TX\text{-}AC\text{-}CM\text{-}PP} = \max(V_{D+} + V_{D-})/2 - \min(V_{D+} + V_{D-})/2$
- $V_{TX\text{-}AC\text{-}CM\text{-}P} = \mathrm{RMS}[(V_{D+} + V_{D-})/2 - DC_{AVG}(V_{D+} + V_{D-})/2]$

Note: The maximum value is calculated on a per unit interval evaluation. The maximum function as described is implicit for all peak-to-peak and peak equations throughout the rest of this chapter, and thus a maximum function will not appear in any following subsequent representations of these equations.

In this section, DC is defined as all frequency components below F_{DC} = 30 kHz. AC is defined as all frequency components at or above F_{DC} = 30 kHz. These definitions pertain to all voltage and current specifications.

An example waveform is shown in Figure 4-35. In this waveform the differential voltage (defined as D+ - D-) is approximately 800 mVPP, and the single-ended voltage for both D+ and D- is approximately 400 mVPP for each. Note that while the center crossing point for both D+ and D- is nominally at 200 mV, the corresponding crossover point for the differential voltage is at 0.0 V.

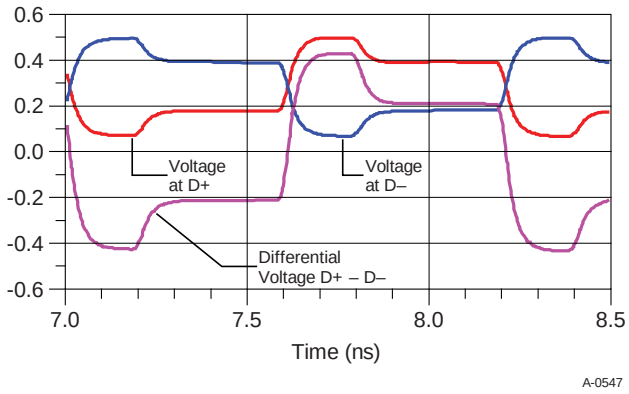


Figure 4-35: Single-ended and Differential Levels

4.3.3.2.1. 2.5 and 5.0 GT/s Specific Voltage Definitions

Voltage and time margins for the Transmitter are defined via a voltage vs. time plot, as illustrated below. The margins are defined as they appear at the test load, located near the Transmitter's pins, or as they appear after the test setup is de-embedded. Voltage swings are referenced to the zero crossover point, since doing so allows the specification to guarantee symmetry. A fixed, two-tap de-emphasis level is stipulated in the specification, where the Transmitter drives to a full swing each time there is a data transition. Otherwise, the data is driven to a partial swing.

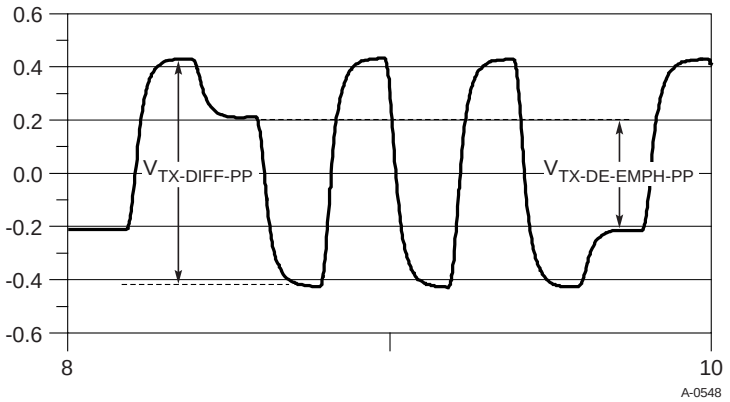


Figure 4-36: Full Swing Signaling Voltage Parameters Showing -6 dB De-emphasis

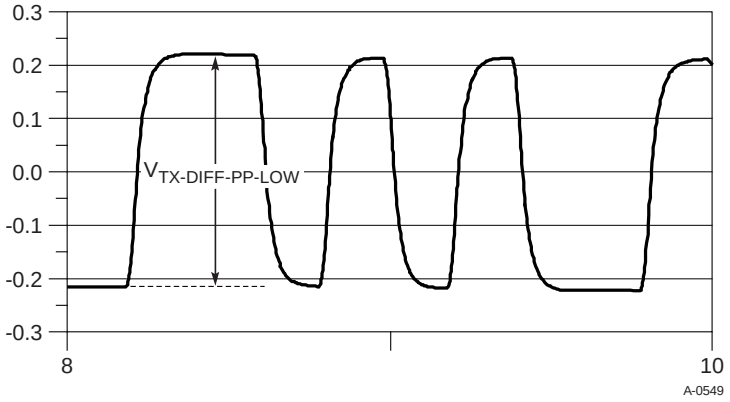


Figure 4-37: Reduced Swing Tx Parameters

A minimum single pulse width $T_{MIN-PULSE}$ is defined in order to limit the amount of channel-induced jitter multiplication. Note that $T_{MIN-PULSE}$ is measured from transition center to the next transition center, and that the transition centers will not always occur at the differential zero crossing point. In particular, transitions from a de-emphasized level to a full level will have a center point offset from the differential zero crossing.

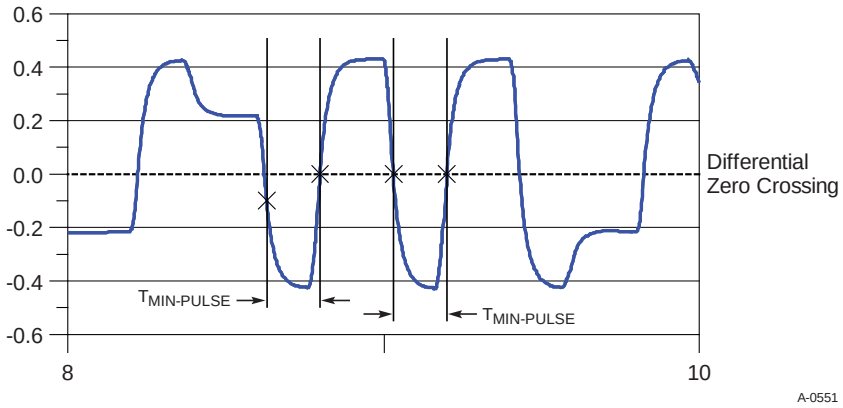


Figure 4-38: Minimum Pulse Width Definition

4.3.3.3. 2.5 and 5.0 GT/s Transmitter De-emphasis

De-emphasis must be implemented when multiple bits of the same logical output state (+1 or -1) are output in succession. Subsequent bits are driven to a differential voltage level as defined by $V_{TX\text{-}DE\text{-}RATIO\text{-}3.5dB}$ and $T_{TX\text{-}DE\text{-}RATIO\text{-}6dB}$ in Table 4-18. At 5.0 GT/s de-emphasis is selectable via various methods including configuration register bits. The two de-emphasis values defined for 5.0 GT/s operation permit optimum equalization for both short, reflection dominated channels, and long, loss dominated ones. Note that all transition bits corresponding to a change in the logical output state (from -1/0 → +1 or +1/0 → -1) must be driven to an amplitude defined by $V_{TX\text{-}DIFF\text{-}PP}$ or $V_{TX\text{-}DIFF\text{-}PP\text{-}LOW}$.

The de-emphasis level is defined via configuration register bits in Chapter 7 and Section 4.2.

The only exception pertains to transmitting the Beacon (see Section 4.3.5.7).

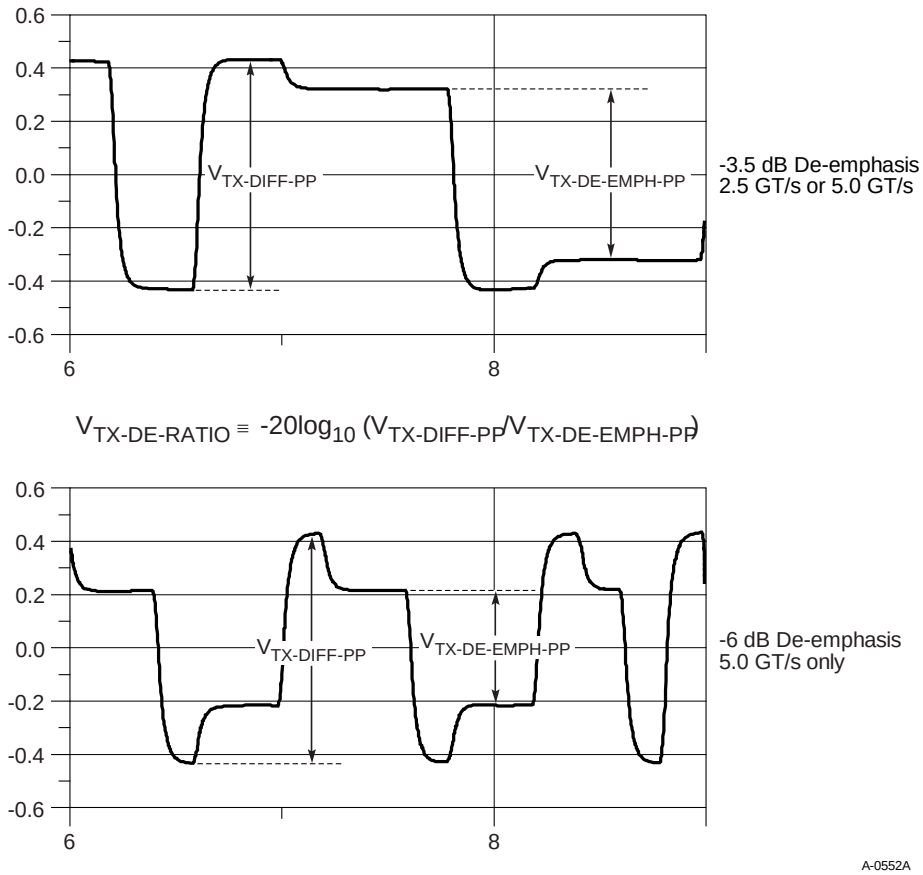


Figure 4-39: Full Swing Tx Parameters Showing De-emphasis

Figure 4-40 illustrates the relationship between a raw data record and an average applied on a per time slice basis. The averaging function generates a single trace from which the de-emphasis ratio may be obtained. Note that the voltage measurements are referenced to the center of each UI. This implies that the measurement must implement transmit phase jitter filtering, either as a real-time capability or as a post processing step, and that the UI center is obtained only after the above-mentioned filtering has been applied.

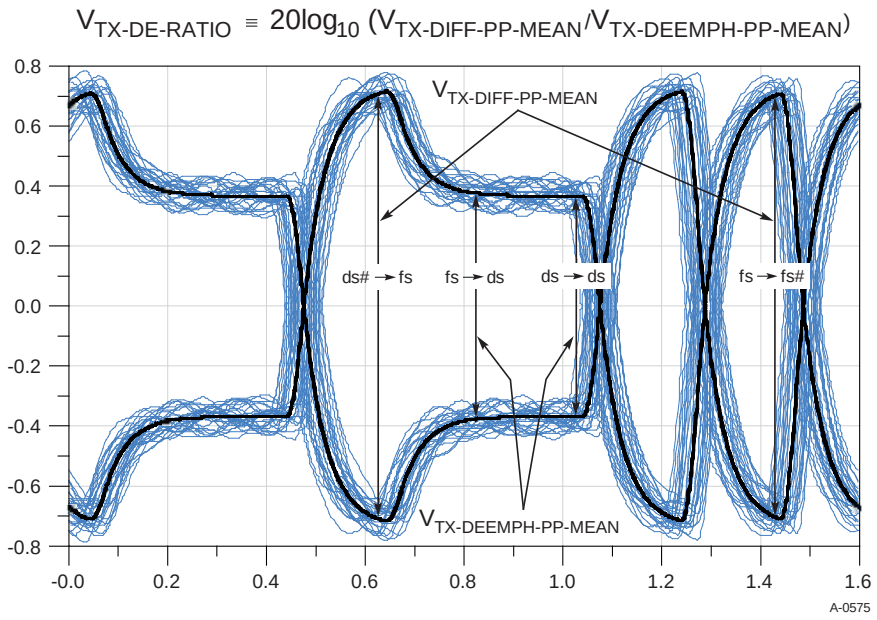


Figure 4-40: Measuring Full Swing/De-emphasized Voltages From Eye Diagram

The above figure is constructed by first generating a data pattern that comprehends all valid transitions between full swing and de-emphasized transitions. These include:

- Full swing to full swing of opposite polarity (fs → fs#)
- Full swing to de-emphasized of same polarity (fs → ds)
- De-emphasized to de-emphasized of same polarity (ds → ds)
- De-emphasized to full swing of opposite polarity (ds → fs#)

A second pattern is also generated using the same transitions, but with complementary data. Each pattern is then averaged over each time slice, and the two averages are overlaid. The resulting diagram yields two measurements each for $V_{TX\text{-}DIFF\text{-}PP\text{-}MEAN}$ and $V_{TX\text{-}DEEMPH\text{-}PP\text{-}MEAN}$. A Transmitter is defined to meet $V_{TX\text{-}DE\text{-}RATIO}$ if all combinations of $-20\log_{10}(V_{TX\text{-}DEEMPH\text{-}PP\text{-}MEAN}/V_{TX\text{-}DIFF\text{-}PP\text{-}MEAN})$ lie within the min-max value of $V_{TX\text{-}DE\text{-}RATIO}$ defined in the specification.

4.3.3.4. Reduced Swing at 2.5 and 5.0 GT/s

Both the 2.5 GT/s and 5.0 GT/s PCI Express specifications define two voltage swing levels: full swing and reduced swing. Typically, reduced swing is specified for power sensitive applications where a shorter channel is acceptable. The requirement as to whether a Transmitter need support full swing, reduced swing, or both modes, is dependent on its usage model. The method by which the output mode is selected is not explicitly defined in this specification, and may be implementation dependent. Note: All PCI Express device Transmitters must support full swing signaling, while support for reduced swing signaling is optional.

While two different Transmitter output signaling levels are defined, only a single Receiver specification is defined; this implies that margins (as specified at the Receiver) are identical regardless of the Transmitter's output swing capabilities. It also implies that the channel's characteristics need to be matched to the Transmitter output swing. Typically, reduced swing output is utilized for short channels, such as would occur in mobile platforms.

4.3.3.5. 8.0 GT/s Tx Voltage Parameters

Tx voltage parameters for 8.0 GT/s include equalization coefficients, equalization presets, and min/max voltage swings.

Tx voltage swing and equalization presets are measured by means of a low frequency pattern within the compliance pattern. Consisting of a sequence of 64 ones followed by 64 zeroes, this pattern permits an accurate measurement of voltage since, ISI effects will have decayed and the signal will approach a steady state.

4.3.3.5.1. 8.0 GT/s Transmitter Equalization

An 8.0 GT/s transmitter shall implement a coefficient-based equalization mode in order to support fine grained control over Tx equalization resolution. Additionally, a transmitter shall support a specified number of presets that give a coarser control over Tx equalization resolution. Both coefficient space and preset space are controllable via messaging from the receiver via an equalization procedure. The equalization procedure operates on the same physical path as normal signaling and is implemented via extensions to the existing protocol link layer.

All transmitters must implement support for the equalization procedure, whereas receivers may optionally implement it. Details of the equalization procedure may be found in Section 4.2.3 of the protocol subsection.

Tx equalization coefficients are based on the following FIR filter relationship as shown in Figure 4-41. Equalization coefficients are subject to constraints limiting their max swing to ±unity with c_{-1} and c_{+1} being zero or negative. The inclusion of the unity condition means that only two of the three coefficients need to be specified to fully define v_out_n. In this specification the two coefficients so specified are c_{-1} and c_{+1}, where c_0 is implied. Note that the coefficient magnitude is not the same as the Tx voltage swing magnitude.

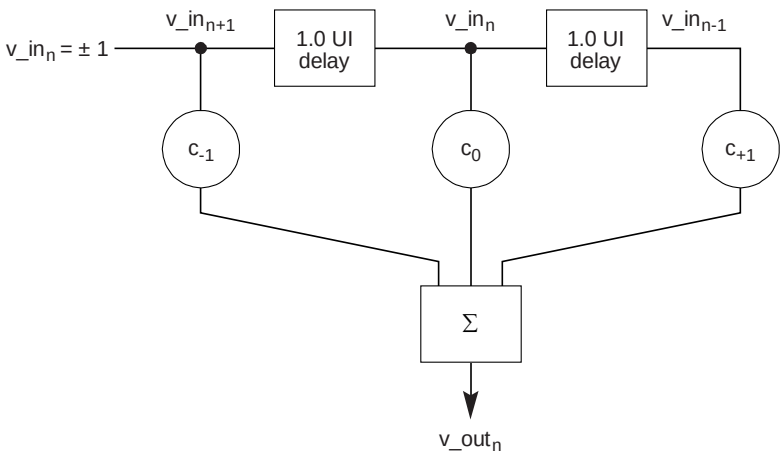


Figure 4-41: Tx Equalization FIR Representation

4.3.3.5.2. Tx Equalization Presets

An 8.0 GT/s transmitter must support a specified number of presets, where the preset values are defined in terms of voltage ratios as illustrated in Figure 4-42. Presets are defined in terms of two ratios, relating the pre-cursor and post-cursor equalization voltages. The pre-cursor (Vc) is referred to as pre-shoot, while the post-cursor (Vb) is referred to as de-emphasis. This convention permits the specification to retain the existing 2.5 GTs/s/5.0 GT/s definitions for Tx equalization, where only de-emphasis is defined, and it allows pre-shoot and de-emphasis to be defined such that each is independent of the other. The maximum swing, Vd, is also shown to illustrate that, when both c_{+1} and c_{-1} are nonzero, the swing of Va does not reach the maximum as defined by Vd. Figure 4-42 is shown as an example of TxEQ and is not intended to represent the signal as it would appear for measurement purposes. The high frequency nature of 8.0 GT/s signaling makes measurement of single UI pulse heights impractical. Consequently all amplitude measurements are made with low frequency waveforms, as shown later in this section.

The presets defined in Table 4-16 and Table 4-17 are numbered to match the designations in Section 4.2.3.1. Note, however, that in Section 4.2.3.1 the coefficients are defined as integers (using upper case "C"), while in the electrical section they are defined as floating point (using lower case "c").

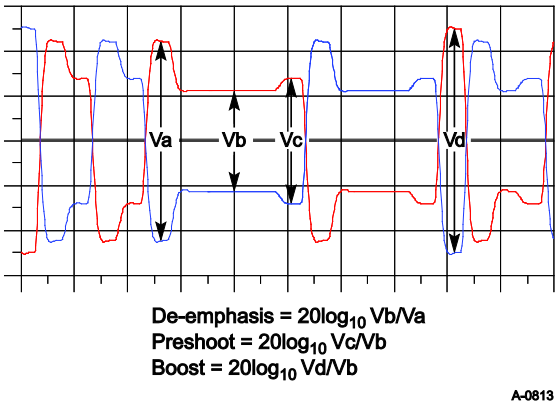


Figure 4-42: Definition of Tx Voltage Levels and Equalization Ratios

Figure 4-43 illustrates the waveforms that will be observed when measuring presets when using the 64-ones/64-zeroes sequence that is part of the compliance pattern. Depending on whether the preset implements de-emphasis, preshoot, or both the corresponding waveshape will differ. The two cases illustrated below show preshoot and de-emphasis, as can be observed by noting whether the boost occurs before or after each edge transition. For the case where both de-emphasis and preshoot are present, boost occurs both before and after each edge transition. Note that in all cases, however, the voltage of interest occurs during the flat portion (Vb) of the waveform, where it can be accurately measured independent of high frequency effects.

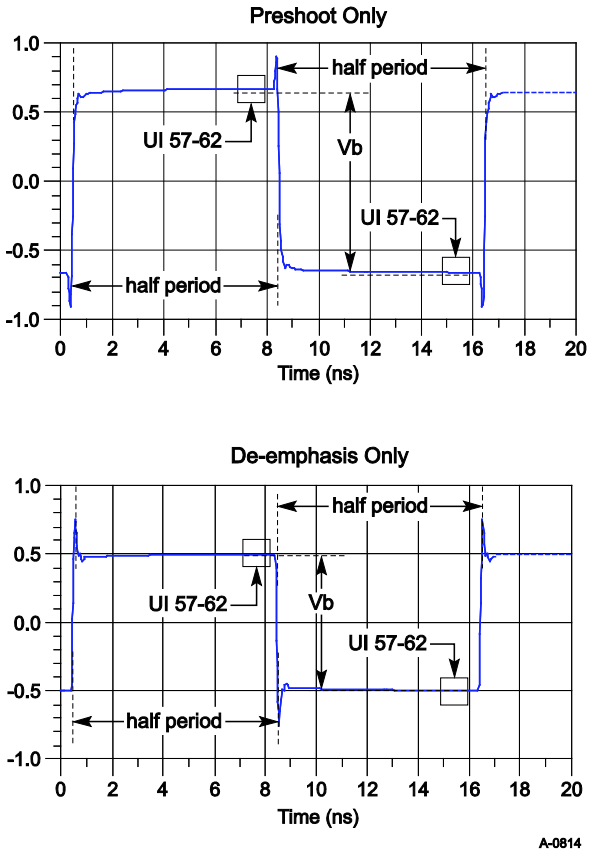


Figure 4-43: Waveform Measurement Points for Pre-shoot and De-emphasis

Table 4-16: Tx Preset Ratios and Corresponding Coefficient Values

Preset Number	Preshoot (dB)	De-emphasis (dB)	c_{-1}	c_{+1}	Va/Vd	Vb/Vd	Vc/Vd
P4	0.0	0.0	0.000	0.000	1.000	1.000	1.000
P1	0.0	-3.5 ± 1 dB	0.000	-0.167	1.000	0.668	0.668
P0	0.0	-6.0 ± 1.5 dB	0.000	-0.250	1.000	0.500	0.500
P9	3.5 ± 1 dB	0.0	-0.166	0.000	0.668	0.668	1.000
P8	3.5 ± 1 dB	-3.5 ± 1 dB	-0.125	-0.125	0.750	0.500	0.750
P7	3.5 ± 1 dB	-6.0 ± 1.5 dB	-0.100	-0.200	0.800	0.400	0.600
P5	1.9 ± 1 dB	0.0	-0.100	0.000	0.800	0.800	1.000
P6	2.5 ± 1 dB	0.0	-0.125	0.000	0.750	0.750	1.000
P3	0.0	-2.5 ± 1 dB	0.000	-0.125	1.000	0.750	0.750
P2	0.0	-4.4 ± 1.5 dB	0.000	-0.200	1.000	0.600	0.600
P10	0.0	See Note 2.	0.000	See Note 2.	1.000	See Note 2.	See Note 2.

Notes:

1. Reduced swing signaling must implement presets #4, #1, #9, #5, #6, and #3. Full swing signaling must implement all the above presets.
2. P10 boost limits are not fixed, since its de-emphasis level is a function of the LF level that the Tx advertises during training. The allowable P10 boost range is defined by the coefficient space lying between the Min and Max Reduced Swing Limits as shown by the two diagonal lines in the example of ~~two diagonal lines in~~ Figure 4-45. This approach permits both full and reduced swing transmitters to use P10 for testing to their respective boost limits.

Comment [VS151]: B14

With the exception of P4 (for which both pre-shoot and de-emphasis are 0.0 dB) it is not possible to obtain a direct measurement of Va and Vc, because these portions of the waveform are 1 UI wide and therefore subject to attenuation by the package and the breakout channel. Instead the Va and Vc values are obtained by setting the DUT to a different preset value where the desired Va or Vc voltage occurs during the Vb interval. Table 4-17 lists the preset values required to measure each of the Va and Vc values from which the preshoot or de-emphasis values may be derived.

Table 4-17: Preset Measurement Cross Reference Table

Preset Number	De-emphasis (dB) $20\log_{10}$ (Vb(i)/Vb(j))	Preshoot (dB) $20\log_{10}$ (Vb(i)/Vb(j))
P4	N/A	N/A
P1	P1/P4	N/A
P0	P0/P4	N/A
P9	N/A	P4/P9
P8	P8/P6	P3/P8
P7	P7/P5	P2/P7
P5	N/A	P4/P5
P6	N/A	P4/P6
P3	P3/P4	N/A
P2	P2/P4	N/A
P10	P10/P4	N/A

4.3.3.5.3. Method for Measuring $V_{TX\text{-}FS\text{-}NO\text{-}EQ}$

The range for a transmitter's output voltage swing, (specified by Vd) with no equalization is defined by $V_{TX\text{-}FS\text{-}NO\text{-}EQ,}$ and is obtained by setting c_{-1} and c_{+1} to zero and measuring the PP voltage on the 64-ones/64-zeroes segment of the compliance pattern. ISI and switching effects are minimized by restricting the portion of the curve over which voltage is measured to the last few UI of each half cycle, as illustrated in Figure 4-44. High frequency noise is mitigated by averaging over multiple readings until the PP noise over the area of interest is less than 2% of the magnitude of $V_{TX\text{-}FS\text{-}NO\text{-}EQ}$. $V_{TX\text{-}FS\text{-}NO\text{-}EQ}$ is time averaged to remove test equipment measurement noise. If the actual Tx has voltage noise that is removed by this averaging process, then the average Tx voltage swing must be increased accordingly.

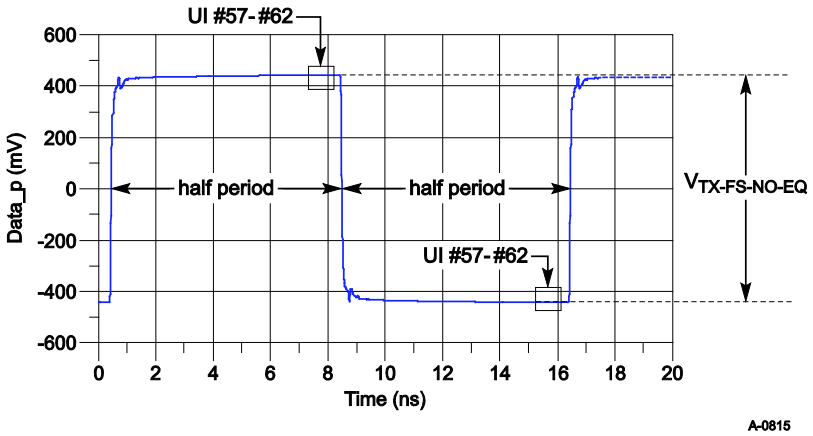


Figure 4-44: $V_{TX\text{-}FS\text{-}NO\text{-}EQ}$ Measurement

4.3.3.5.4. Coefficient Range and Tolerance

An 8.0 GT/s transmitter is required to inform the receiver of its coefficient range and tolerance, and this requirement relaxes the constraints that the electrical specification must impose on coefficients. Coefficient range and tolerance are constrained by the following requirements. For details see Section 4.2.3.1 in the protocol subsection.

- Coefficients must support all eleven presets and their respective tolerances as defined in Table 4-16.
- All transmitters must meet the full swing signaling $V_{TX\text{-}EIEOS\text{-}FS}$ limits.
- Transmitters may optionally support reduced swing, and if they do, they must meet the $V_{TX\text{-}EIEOS\text{-}RS}$ limits.
- The coefficients must meet the boost and resolution ($V_{TX\text{-}BOOST\text{-}FS}$, $V_{TX\text{-}BOOST\text{-}RS}$ and $EQ_{TX\text{-}COEFF\text{-}RES}$) limits defined in Table 4-19.

When the above constrains are applied the resulting coefficient space may be mapped onto a triangular matrix, an example of which is shown in Figure 4-45. The matrix may be interpreted as follows: Pre-shoot and de-emphasis coefficients are mapped onto the Y-axis and X-axes, respectively. In both cases the maximum granularity of 1/24 is assumed. Each matrix cell corresponding to a valid combination of preshoot and de-emphasis coefficients has three entries corresponding to preshoot (PS), de-emphasis (DE), and boost (as shown in the upper left hand corner). Diagonal elements are defined by the maximum boost ratio. Those cells highlighted in blue are supported for reduced swing, while cells in either blue or orange represent presets supported for full swing signaling. Note that this figure is informative only and is not intended to imply any particular Tx implementation.

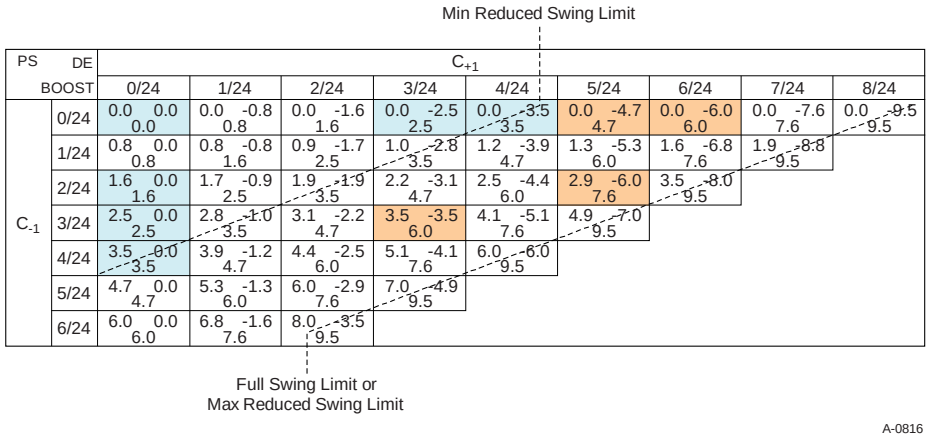


Figure 4-45: TxEQ Coefficient Space Triangular Matrix Example

4.3.3.5.5. EIEOS Pattern and $V_{TX\text{-}EIEOS\text{-}FS}$ and $V_{TX\text{-}EIEOS\text{-}RS}$ Limits

$V_{TX\text{-}EIEOS\text{-}FS}$ and $V_{TX\text{-}EIEOS\text{-}RS}$ are measured using the EIEOS sequence contained within the compliance pattern. This pattern consists of eight consecutive ones followed by the same number of consecutive zeroes, where the pattern is repeated for a total of 128 UI. The loss effect of the breakout channel may be appreciable at the EIEOS signaling frequency, so its loss effects must be taken into account to yield an equivalent voltage at the Tx pin. Typically this requires de-embedding.

A transmitter sends out a unique EIEOS pattern to inform the receiver that the transmitter is signaling an EI Exit. This pattern guarantees the receiver will properly detect the EI Exit condition, something not guaranteed by scrambled data. The Tx EIEOS launch voltage is defined by $V_{TX\text{-}EIEOS\text{-}FS}$ for full swing signaling and by $V_{TX\text{-}EIEOS\text{-}RS}$ for reduced swing signaling. $V_{TX\text{-}EIEOS\text{-}RS}$ is smaller than $V_{TX\text{-}EIEOS\text{-}FS}$ to reflect the fact that reduced swing is typically supported only for lower loss channels where there is less attenuation at the EIEOS signaling rate.

For full swing signaling $V_{TX\text{-}EIEOS\text{-}FS}$ is measured with a preset number P10. This is equivalent to a maximum nominal boost of 9.5 dB and represents the maximum boost attainable in coefficient space. When a boost tolerance of ±1.5 dB is factored in this yields the minimum boost limit of 8.0 dB appearing in Table 4-19. For reduced swing signaling $V_{TX\text{-}EIEOS\text{-}RS}$ is measured with preset P1.

Both $V_{TX\text{-}EIEOS\text{-}FS}$ and $V_{TX\text{-}EIEOS\text{-}RS}$ are referenced to the Tx pin, so any attenuation effects of the breakout channel must be removed from the measurement, typically by de-embedding. Figure 4-46 illustrates the de-emphasis peak as observed at the pin of a Tx for $V_{TX\text{-}EIEOS\text{-}FS}$. At the far end of a lossy channel the de-emphasis peak will be attenuated; this is why the measurement interval includes only the middle five UI. The voltage is averaged over this interval for both the negative and positive halves of the waveform. $V_{TX\text{-}EIEOS\text{-}FS}$ or $V_{TX\text{-}EIEOS\text{-}RS}$ is defined as the difference between the negative and positive waveform segment averages. UI boundaries are defined with respect to the edge of the recovered data clock.

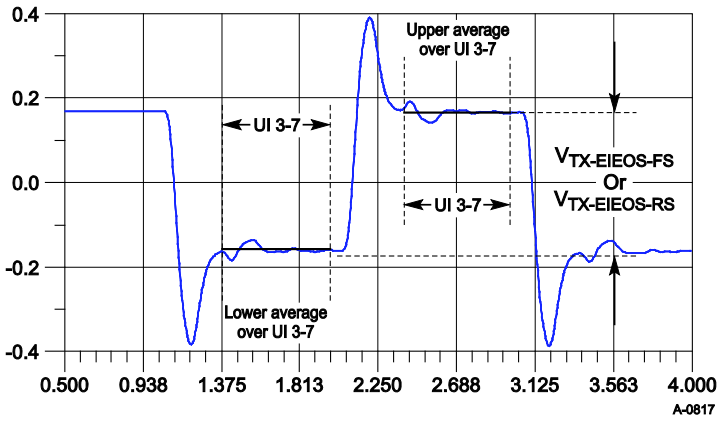


Figure 4-46: Measuring $V_{TX\text{-}EIEOS\text{-}FS}$ and $V_{TX\text{-}EIEOS\text{-}RS}$

4.3.3.5.6. Reduced Swing Signaling

8.0 GT/s PCI Express transmitters may support a reduced swing signaling in a manner similar to that defined for 5.0 GT/s. $V_{TX\text{-}RS\text{-}NO\text{-}EQ}$ has the same maximum of 1200 mVPP as $V_{TX\text{-}FS\text{-}NO\text{-}EQ}$. Its minimum swing is captured indirectly by $V_{TX\text{-}EIEOS\text{-}RS}$, so there is no need to define a separate minimum limit for $V_{TX\text{-}RS\text{-}NO\text{-}EQ}$. Operating at a reduced swing limits the range of presets and the maximum boost; the maximum boost (Vd/Vb) supported for reduced swing is 3.5 dB.

4.3.3.5.7. Effective Tx Package Loss

Package loss (including silicon driver bandwidth) is defined by the $ps21_{TX}$ parameter. Since both package and driver characteristics affect the signal as observed at the Tx pin, this parameter has the advantage of capturing both effects while permitting the measurement to be made at a point (TP1) that can easily be probed. It is necessary to include a package loss parameter in the Tx specification, since the voltage swing ($V_{TX\text{-}FS\text{-}NO\text{-}EQ}$) is defined at an equivalent pulse frequency of 1/128 UI or 62.5 MHz, a frequency chosen specifically to minimize package loss effects.

Package loss is measured by comparing the 64-ones/64-zeroes PP voltage (V_{111}) against a 1010 pattern (V_{101}). Tx package loss measurement is made with c_{-1} and c_{+1} both set to zero. Multiple measurements shall be made and averaged to obtain stable values for V_{101} and V_{111}. Due to the HF content of V_{101}, $ps21_{TX}$ measurement requires that the breakout channel be de-embedded back to the Tx pin.

Note that the compliance pattern for Lanes other than Lane 0 will differ slightly from what is shown in Figure 4-47. For details refer to Section 4.2.10 ff.

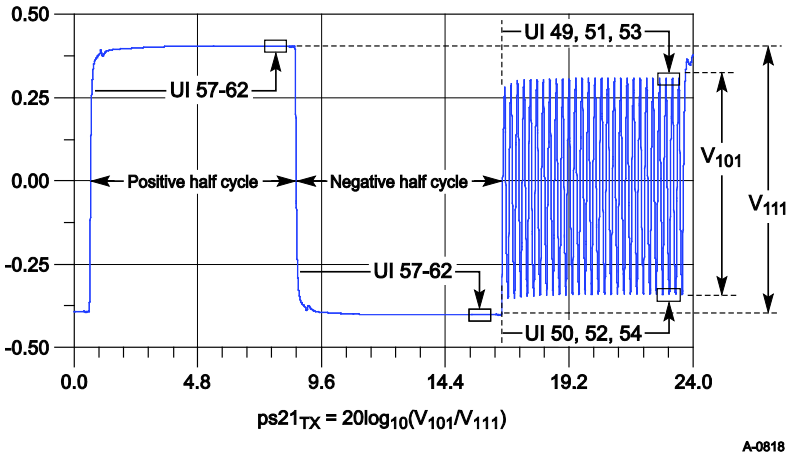


Figure 4-47: Compliance Pattern and Resulting Package Loss Test Waveform

Measurement of V_{101} and V_{111} is made towards the end of each interval to minimize ISI and low frequency effects. V_{101} is defined as the peak-peak voltage between minima and maxima of the clock pattern. V_{111} is defined as the peak-peak voltage difference between the positive and negative levels of the two half cycles. The measurement should be averaged over multiple compliance patterns until the mean deviates by less than 2% between successive averages.

4.3.3.6. 5.0 GT/s Transmitter Margining

5.0 GT/s Transmitters must implement voltage margining for both 5.0 GT/s and 2.5 GT/s operation, while 2.5 GT/s only Transmitters need not implement voltage margining. When operating in the margining mode, a Transmitter outputs a signal at an amplitude whose value is determined by configuration register bits. The base specification only requires that a Tx be capable of margining voltage. However, other parameters, such as jitter and/or de-emphasis level may be optionally margined. Figure 4-48 illustrates the Tx margining voltages and corresponding margining configuration register codes.

The granularity for margining control is determined by the number of Lanes per Link such that margining must be controllable on at least a per Link basis and may be controllable on a per Lane basis.

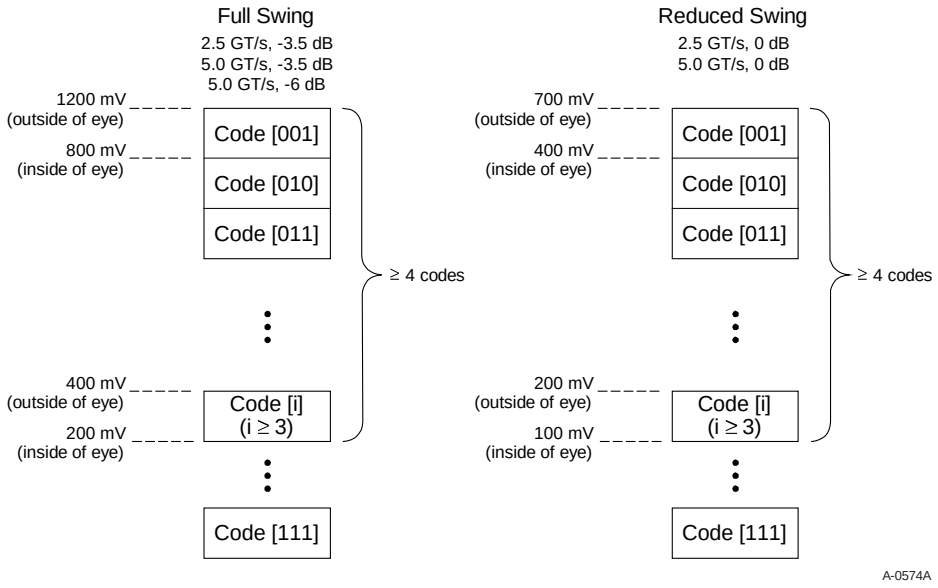


Figure 4-48: Transmitter Margining Voltage Levels and Codes

4.3.3.7. 8.0 GT/s Transmitter Margining

8.0 GT/s transmitters shall implement a margining procedure that allows the Tx launch voltage to be adjusted. Margining is enabled by programming the same register set as defined for 5.0 GT/s. Due to the larger range of TxEQ 8.0 GT/s Tx margining is subject to additional constraints: Tx margining shall not require any coefficient or preset resolution finer than can be generated with 1/24 coefficient resolution defined for normal operation, and shall not require more Tx accuracy or capability than is required to support normal operation. It is acceptable that Vb fall below the limit set by $V_{TX\text{-}EIEOS\text{-}FS}$ or $V_{TX\text{-}EIEOS\text{-}RS}$, although proper end to-end operation is no longer guaranteed.

The implied accuracy of TxEQ that can be implemented with an 8.0 GT/s Tx also applies to 2.5 GT/s and 5.0 GT/s operation.

4.3.3.8. Jitter Budgeting and Measurement

PCI Express jitter is budgeted among the components that comprise an end-to-end connection: the Refclk, Transmitter and receiver. Channel jitter is indirectly specified by requiring specific jitter and voltage margins at the far end of the channel under test into a reference load.

Deterministic jitter peak-peak terms are added arithmetically, while random jitter sigma terms are added as the square root of sum of squares (RSS) and its peak-peak equals the overall sigma after RSS multiplied by the Qfactor at the interested BER level as shown in Equation 4.3.1. For example, a 10^{-12} BER corresponds to $Q_{BER} = \pm 7.03$.

$$System\ Tj \equiv \sum Dj + 2Q_{BER}\sqrt{\sum Rj^2} \le 1.0UI \qquad \textbf{Equation 4.3.1}$$

A consequence of the RSS adding of Rj terms is that naively adding the Tj terms for Refclk, Transmitter, and Receiver will yield a sum greater than 1.0 UI. Details on how to derive meaningful jitter numbers from measurements appear in the respective sections for Transmitter, Receiver, and Refclk.

Jitter may be resolved into separate Dj and Rj components by application of the Dual Dirac approximation method. The dual-Dirac model is a technique for quickly estimating total jitter defined at a low bit error ratio, Tj(BER), while not requiring a commensurately large number of samples. The deterministic and random components of the jitter signal are separated within the context of the model to yield two quantities, root-mean-square random jitter (Rj) and a model-dependent form of the peak-to-peak deterministic jitter, Dj(dd). The dual-Dirac model is fundamentally a Gaussian approximation to the outer edges of the jitter distribution displaced by a fixed amount, Dj(dd). It rests on the assumption that the asymptotic tails of the distribution may be represented by the tails of the same Gaussian distribution that describes Rj; this is the key to how the dual-Dirac model is used to calculate Tj(BER): measurements of Rj and Dj(dd) may be performed on comparatively low statistics and Tj(BER) can be calculated using Equation 4.3.2.

$$Tj(BER) = 2Q_{BER} \times \sigma + Dj(dd) \qquad \textbf{Equation 4.3.2}$$

It is important to realize that Tj(BER) calculated from Rj and Dj(dd) is not a model-dependent approximation. Its accuracy depends only on the accuracy with which Rj and Dj(dd) are measured – though, in practice Tj(BER) so calculated is almost always an extrapolation. While Dj(dd) is a model dependent quantity, it is also a well defined observable. On the other hand, Rj is not a model-dependent quantity; it is the width of the Gaussian distribution that describes the random jitter PDF.

It is important to realize that Dj(dd) is not the same as the actual peak-to-peak spread of the Dj distribution, Dj(p-p). The two are related by Dj(dd) ≥ Dj(p-p). Dj(dd) has two distinct advantages over Dj(p-p): first, it is useful in estimating Tj(BER) – substitution of Dj(p-p) for Dj(dd) in Eq. (4.3.1) is incorrect; and, second, Dj(dd) is much easier to measure.

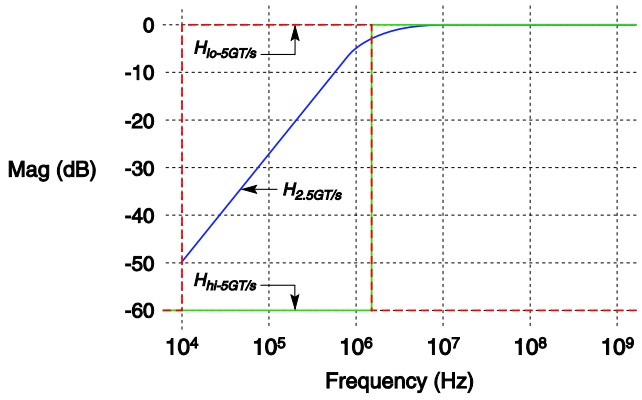
Transmitter AC performance is specified at pins of the DUT into a reference load, as illustrated in Figure 4-31. The data so obtained consists of a sequential set of voltage vs. time samples. Most Transmitter parameters may be obtained directly from the scope trace; however, some parameters must either be post processed or filtered directly to obtain meaningful eye margins. Either approach is designed to remove jitter components that would be tracked by the Receiver and do not contribute meaningfully to the effective jitter seen at the Receiver's input latch.

Note that the Transmitter under test is driven by a low jitter "ideal" clock source. An ideal, rather than a nominal, oscillator is specified so that the measured jitter represents only those contributions from the Transmitter under test. The reference clock has a jitter budget that is specified separately.

4.3.3.9. 2.5 GT/s and 5.0 GT/s Transmitter Phase Jitter Filtering

Tx phase jitter is defined as the temporal deviation of a data edge from an ideal clock in a manner similar to the Refclk phase jitter as described in the JWG white paper. As illustrated in Figure 4-31, a Transmitter is characterized to specification while being driven by an "ideal", low jitter Refclk. Such a Refclk source contributes no appreciable jitter of its own within the Transmitter's PLL passband. Therefore, the jitter observed at the Transmitter's output can be assumed to be entirely generated by the DUT. However, the jitter so measured may still contain low frequency jitter components that would be tracked by a Receiver and should not be included in the measurement. Examples include oscillator and measurement apparatus baseline wander. It is necessary to process the raw jitter to remove this low frequency jitter, where this may be implemented either via software (as a post processing step) or via hardware (as an actual filter). In either case, the algorithm must emulate the filter functions as defined below.

2.5 GT/s specifies a 1-pole HPF with a 1.5 MHz corner frequency, while 5.0 GT/s specifies a step bandpass filter with lower and upper corner frequencies of 10 kHz and 1.5 MHz, respectively. For filter details, see Figure 4-49.



$$H_{2.5GT/s} = \frac{s}{s + w_c} \qquad w_c = 2\pi f_T$$

$H_{hi\text{-}5GT/s} = if(f >= f_T)$ *then* 1.0 *else* 10^{-3}

$H_{lo\text{-}5GT/s} = if(f < f_{10kHz})$ *then* 10^{-3}

elseif $(f < f_T)$ *then* 1.0

else 10^{-3}

$f_T = 1.5$ *MHz*

A-0544

Figure 4-49: Plot of Transmitter HPF Filter Functions

4.3.3.9.1. Measuring Tx Eye Width in the Presence of De-Emphasis

When a measurement is taken at the Tx pins, de-emphasis induces a jitter artifact that must be removed in order to obtain a meaningful jitter number. The origin of this inaccuracy lies in the finite rise/fall time of the Transmitter and the unequal voltage swings between different transitions. De-emphasis jitter must be accounted for in specifying 5.0 GT/s signaling and may be accounted for at 2.5 GT/s.

De-emphasis induced jitter artifacts may be removed by the algorithm listed below. Consisting solely of scale and shift operations, the algorithm operates exclusively in the voltage domain. Its effect on a typical Tx signal is illustrated in Figure 4-51. The high amplitude eye represents the jitter that would be observed without removing de-emphasis artifacts, while the low amplitude eye shows the effect of applying the above algorithm. Other algorithms yielding identical results may also be used.

Note that the above algorithm yields only a correct eye width. Eye voltage should be obtained from Tx waveforms as shown in Figure 4-40. The *deemp* parameter listed below in the algorithm is obtained empirically from the data record as illustrated in Figure 4-51 and will vary depending on whether a nominal de-emphasis of -3.5 dB or -6 dB is applied.

$$
\begin{aligned}
&IF\ fullswing: \\
&\quad scale = deemp; \quad offset = 0.0 \\
&ELSEIF\ \ previousBit == 1: \\
&\quad scale = \frac{2 \cdot deemp}{1 + deemp}; \quad offset = \frac{1 - scale}{4} \\
&ELSE: \\
&\quad scale = \frac{2 \cdot deemp}{1 + deemp}; \quad offset = -\frac{1 - scale}{4}
\end{aligned}
$$

Figure 4-50: Algorithm to Remove De-Emphasis Induced Jitter

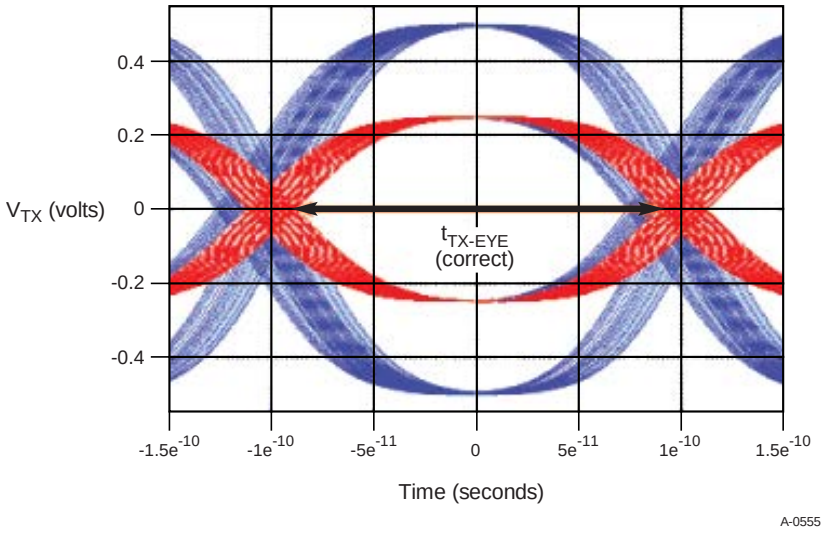


Figure 4-51: Example of De-emphasis Jitter Removal

At 5.0 GT/s it is necessary to resolve jitter into Tj and Dj components as defined by T_{TX-EYE} and $T_{TX-DJ-DD}$, respectively. In addition, another parameter, $T_{MIN-PULSE}$ is required to place a lower limit on a single pulse width. Measurement of T_{TX-EYE} at 2.5 GT/s and 5.0 GT/s is essentially identical with the exception that a different filter function is applied at the two bit rates (see Figure 4-49). $T_{TX-DJ-DD}$ may be obtained by applying the Dual Dirac jitter estimation technique.

4.3.3.10. 8.0 GT/s Tx Jitter Parameters

Tx jitter is defined in terms of parameters as listed below in Table 4-19. Jitter is measured at the zero crossing point at the full speed of 8.0 GT/s. When measuring a particular Tx lane it is necessary to ensure that all other PCI Express Lanes are transmitting in order to capture Tx die and package crosstalk effects. When measuring Tx jitter it is required for the DUT to drive as many of its outputs as would occur during normal operation in a system environment.

4.3.3.10.1. Data Dependent and Uncorrelated Jitter

Measured at TP1 and de-embedded back to the pin, a transmitter's jitter contains data dependent and uncorrelated components. The data dependent components occur principally due to package loss and reflection. Uncorrelated jitter sources include PLL jitter, power supply noise, and crosstalk. This specification separates jitter into uncorrelated and data dependent bins, because such a separation matches well with the Tx and Rx equalization capabilities. Uncorrelated jitter is not mitigated by Tx or Rx equalization and represents timing margin that cannot be recovered with equalization. It is important that margin that is recoverable via equalization (data dependent) is not budgeted as non-recoverable jitter.

Once data dependent jitter has been removed from the Tx measurement it becomes possible to resolve the remaining jitter into Tj and DJDD components. High frequency jitter amplification in the channel is accounted for by separate $T_{TX-UPW-DJDD}$ and $T_{TX-UPW-TJ}$ parameters.

4.3.3.10.2. Post Processing Steps to Extract Jitter

Measured Tx jitter is referenced to the Tx pin, and depending on what type of jitter is defined is subsequently referenced either to a recovered data clock or to a data edge. Data captured at TP1 requires post processing in order to remove the effects of the breakout channel and to regenerate a data clock.

4.3.3.10.3. De-embedding

Direct probing at a transmitter's pins is not generally feasible, so data is instead measured at TP1 of the breakout channel. By means of the replica channel it is possible to determine the loss vs. frequency characteristics of the breakout channel and de-embed this channel. The loss effects of the breakout channel may be minimized by means of de-embedding. Note that since de-embedding amplifies HF channel and measurement noise, an HF cutoff limit of between 8 and 12 GHz must be applied to de-embedding. This limit also sets the upper maximum frequency range required for test equipment.

4.3.3.10.4. Behavioral CDR and Clock Recovery Algorithm

Any phase drift, either due to test equipment or the clock source, must be removed to account for the ability of the receiver to track this jitter and will be implemented with a behavioral CDR function. The behavioral CDR shall match the Sj mask's jitter tracking capability of as defined in the Receiver section, Figure 4-74, except that the SJ magnitude shall continue to increase below 1 MHz with a 20 dB/decade slope.

4.3.3.10.5. Data Dependent Jitter ($T_{TX\text{-}DDJ}$)

The resolution of transmitter jitter into constituent components is dependent on the ability to separate data dependent jitter from other jitter types. Separation of jitter into data dependent and uncorrelated components may be achieved by averaging techniques; for example, by having the Tx repeatedly drive the compliance test pattern which is a repeating pattern.

Figure 4-52 illustrates the relation between Tx data, recovered clock, and the data's PDF. Data dependent jitter is defined as the time delta between the PDF's mean for each zero crossing point and the corresponding recovered clock edge. In the example below DDJ1 is negative, DDJ2 is positive, and DDJ3 is zero. A sufficient number of repeated patterns must be accumulated to yield a stable mean value for each edge. $T_{TX\text{-}DDJ}$ is defined as the absolute value of the difference between DDJ(max) – DDJ(min). This method of removing DDJ accomplishes the same goal as was done in the 5.0 GT/s Tx specification without having to perform a scale/shift operation to remove TxEq jitter.

At 8.0 GT/s package-die interactions preclude simultaneous resolution of both voltage and jitter at signaling speed. The 8.0 GT/s Tx specification avoids this problem by measuring TxEq voltages at a much reduced effective bit rate while measuring jitter at full speed. Analysis has shown that jitter can be de-embedded with good accuracy while voltage cannot.

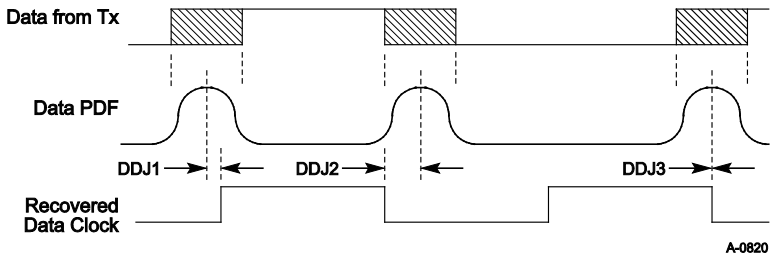


Figure 4-52: Relation Between Data Edge PDFs and Recovered Data Clock

4.3.3.10.6. Uncorrelated Total Jitter and Deterministic Jitter (T_{TX-UTJ} and $T_{TX-UDJDD}$)

This type of jitter is referenced to a recovered data clock generated by means of a CDR tracking function. Uncorrelated jitter may be derived after removing the DDJ component from each PDF and combining the PDFs for all edges in the pattern. By appropriately converting the PDF to a Q-scale it is possible to obtain the graphical relation shown in Figure 4-53, from which T_{TX-UTJ} and $T_{TX-UDJDD}$ may be derived. In Figure 4-53 note that the two PDF curves are identical but that the fitted slopes, defined by $1/RJ_{LH}$ and $1/RJ_{RH}$, may differ.

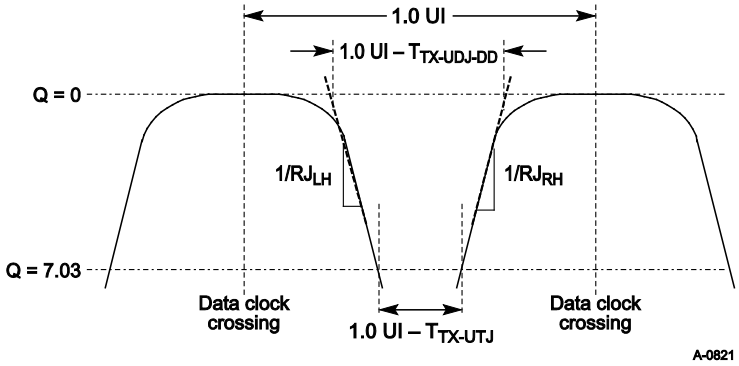


Figure 4-53: Derivation of T_{TX-UTJ} and $T_{TX-UDJDD}$

4.3.3.10.7. Uncorrelated Total and Deterministic PWJ ($T_{TX\text{-}UPW\text{-}TJ}$ and $T_{TX\text{-}UPW\text{-}DJDD}$)

Pulse width jitter is defined as an edge to edge phenomenon on consecutive edges nominally 1.0 UI apart. Figure 4-54 illustrates how PWJ is defined, showing that it is typically present on both data edges of consecutive UI. To accurately quantify PWJ it is first necessary to remove the ISI contributions to PWJ. The shaded areas on either side of the unjittered edges represent the maximum amount of jitter about that edge. Note the jitter for one edge is assumed to be independent from the other.

An equivalent description of PWJ may be obtained by referencing to a fixed leading edge and having jitter contributions from both edges appear at the trailing edge. This approach yields a single PDF as shown below. Each 1 UI pulse in the pattern will have a different median for this PDF which is caused by ISI and F/2 jitter. The average of the medians for 1 UI pulses at odd and even UI numbers within the pattern are calculated, and the odd and even PDF's are normalized to the appropriate average of medians and summed to form an odd UI PDF and an even UI PDF. The final PDF is calculated from the sum of the summed odd and even UI PDF's. The key idea here is that the final PDF for UPWJ should include F/2 or odd/even UI jitter

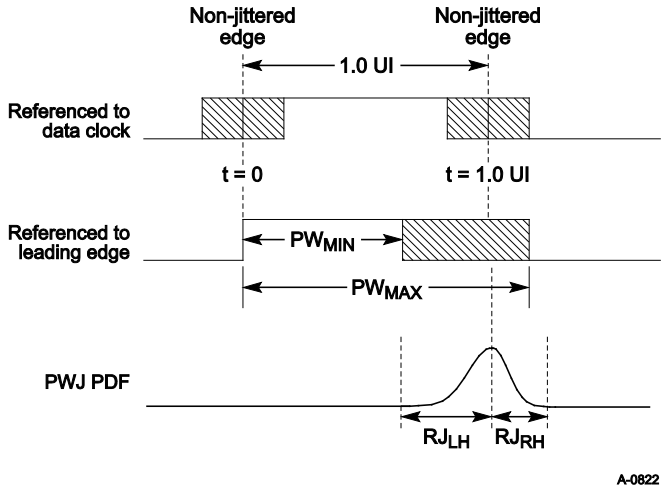


Figure 4-54: PWJ Relative to Consecutive Edges 1 UI Apart

The PDF of jitter around each non-jittered edge may be converted into the Q-scale (see Figure 4-55) from which $T_{TX\text{-}UPW\text{-}TJ}$ and $T_{TX\text{-}UPW\text{-}DJDD}$ may be derived in a manner analogous to $T_{TX\text{-}UTJ}$ and $T_{TX\text{-}UDJDD}$. Note that the PDF may not be symmetric and the tail of interest is RJ_{LH}, since it represents pulse compression.

To obtain the PDF curve shown in Figure 4-55 take the left hand side of the PWJ PDF curve (RJLH), copy and mirror it about the t=1.0 UI axis. The resulting curve shown in Figure 4-55 may be used to compute $T_{TX\text{-}UPW\text{-}TJ}$ and $T_{TX\text{-}UPD\text{-}DJDD}$ at Q = 7.03 and Q = 0, respectively.

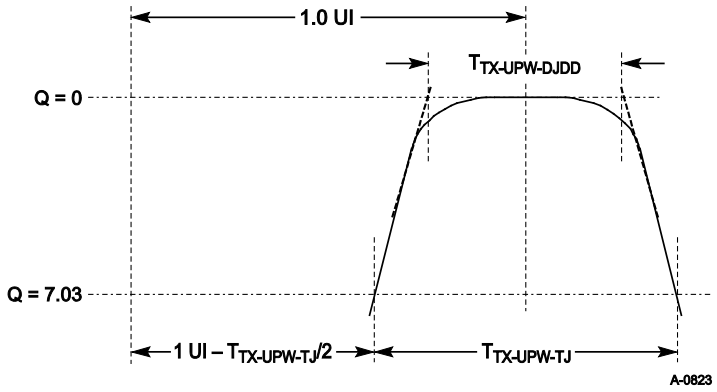


Figure 4-55: Definition of $T_{TX-UPW-DJDD}$ and $T_{TX-UPW-TJ}$

4.3.3.11. Tx and Rx Return Loss

Return loss measurements for the Tx and Rx are essentially identical, so both are included in the transmitter section. Return loss measurements are made at the end of the respective breakout channels and require that the breakout channel's contribution to RL be de-embedded, thereby associating the return loss with the Tx or Rx pin. Figure 4-56 defines the pass/fail mask for differential return loss. There is a small difference in the 2.5-4.0 GHz mask value range to account for the slightly lower C_{DIE} of the receiver as compared to the transmitter. This difference is also reflected in the Tx and Rx behavioral package models. Both differential and common mode are defined over a frequency range of 50 MHz to 4.0 GHz.

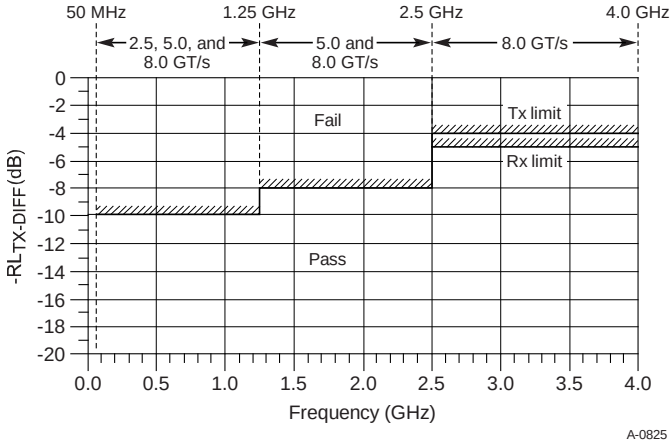


Figure 4-56: Tx, Rx Differential Return Loss Mask

The pass/fail mask for common mode return loss is shown below. Return loss measurements require that both the Tx and Rx are powered up and that their respective termination circuits are enabled.

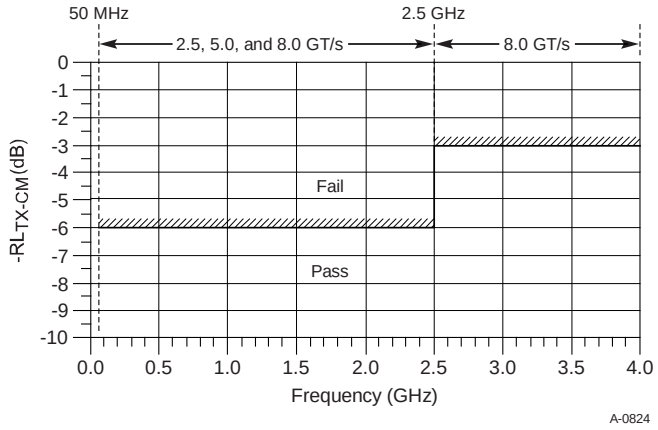


Figure 4-57: Tx, Rx Common Mode Return Loss Mask

4.3.3.12. Transmitter PLL Bandwidth and Peaking

4.3.3.12.1. 2.5 GT/s and 5.0 GT/s Tx PLL Bandwidth and Peaking

PLL bandwidth and peaking are defined for both the Transmitter and Receiver in order to place an upper limit on the amount of Refclk jitter that is propagated to the transmitted data and to the CDR. Defining PLL BW and peaking limits also guarantees a minimum degree of Tx/Rx jitter tracking in those systems utilizing a common Refclk Rx architecture.

Two sets of bandwidth and peaking are defined for 5.0 GT/s: 8-16 MHz with 3 dB of peaking and 5-16 MHz with 1 dB of peaking. This gives the designer the option of trading off between a low peaking PLL design vs. a low bandwidth design. For 2.5 GT/s, a single PLL bandwidth and peaking range is specified at 1.5-22 MHz with 3.0 dB of peaking.

4.3.3.12.2. 8.0 GT/s Tx PLL Bandwidth and Peaking

The Tx and Rx PLL bandwidth for 8.0 GT/s signaling is 2 – 5 MHz. Peaking may be from 0 to 2 dB for bandwidths up to 4 MHz and 0 to 1 dB for bandwidths up to 5 MHz. The 8.0 GT/s PLL BW range is substantially lower than the PLL bandwidths specified for 5.0 GT/s or 2.5 GT/s operation to reduce the amount of Refclk jitter at the sample latch of the receiver.

4.3.3.12.3. Series Capacitors

8.0 GT/s PCI Express requires series capacitors to provide a DC block between Tx and Rx. The min/max capacitance spread has been decreased from that of the 2.5 and 5.0 GT/s standards, while the maximum value has been slightly increased. This change is necessary to minimize DC wander effects due to data scrambling. Note that 2.5 GT/s and 5.0 GT/s signaling must also propagate through these larger value capacitors, but the small increase in capacitor size has no adverse impact on either 2.5 GT/s or 5.0 GT/signaling or low frequency in-band signaling such as receiver detect.

4.3.3.13. Common Transmitter Parameters

The following table defines the parameters for transmitters that are common among all three data rates. Parameters are defined separately for 2.5 GT/s, 5.0 GT/s, and 8.0 GT/s implementations.

Table 4-18: Transmitter Specifications

Symbol	Parameter	2.5 GT/s	5.0 GT/s	8.0 GT/s	Units	Comments
UI	Unit Interval	399.88 (min) 400.12 (max)	199.94 (min) 200.06 (max)	124.9625 (min) 125.0375 (max)	ps	The specified UI is equivalent to a tolerance of ±300 ppm for each Refclk source. Period does not account for SSC induced variations. See Note 1.
BW_{TX-PLL}	Tx PLL BW for 2.5 GT/s	22 (max) 1.5 (min)	Not specified	Not specified	MHz	See Note 6.
$BW_{TX-PKG-PLL1}$	Tx PLL bandwidth corresponding to $PKG_{TX-PLL1}$	Not specified	16 (max) 8 (min)	4 (max) 2 (min)	MHz	Second order PLL jitter transfer bounding function. See Note 6.
$BW_{TX-PKG-PLL2}$	Tx PLL bandwidth corresponding to $PKG_{TX-PLL2}$	Not specified	16 (max) 5 (min)	5 (max) 2 (min)	MHz	Second order PLL jitter transfer bounding function. See Note 6.
$PKG_{TX-PLL1}$	Tx PLL peaking	Not specified	3.0 (max)	2.0 (max)	dB	PLL BW = 8 MHz (min) @ 5.0 GT/s or BW = 4 MHz (max) @ 8.0 GT/s. See Notes 6 and 8.
$PKG_{TX-PLL2}$	Tx PLL peaking	Not specified	1.0 (max)	1.0 (max)	dB	PLL BW = 5 MHz (min) @ 5.0 GT/s or BW = 5 MHz (max) @ 8.0 GT/s. See Note 8.
$V_{TX-DIFF-PP}$	Differential p-p Tx voltage swing	0.8 (min) 1.2 (max)	0.8 (min) 1.2 (max)	See Table 4-19.	VPP	As measured with compliance test load. Defined as $2*\|V_{TXD+} - V_{TXD-}\|$.
$V_{TX-DIFF-PP-LOW}$	Low power differential p-p Tx voltage swing	0.4 (min) 1.2 (max)	0.4 (min) 1.2 (max)	See Table 4-19.	VPP	As measured with compliance test load. Defined as $2*\|V_{TXD+} - V_{TXD-}\|$. See Note 9.
$V_{TX-DE-RATIO-3.5dB}$	Tx de-emphasis level ratio	3.0 (min) 4.0 (max)	3.0 (min) 4.0 (max)	See Table 4-19.	dB	See Section 4.3.3.3 and Note 11 for details.
$V_{TX-DE-RATIO-6dB}$	Tx de-emphasis level	N/A	5.5 (min) 6.5 (max)	See Table 4-19.	dB	See Section 4.3.3.3 and Note 11 for details.
$T_{MIN-PULSE}$	Instantaneous lone pulse width	Not specified	0.9 (min)	See Table 4-19.	UI	Measured relative to rising/falling pulse. See Notes 2, 10, and Figure 4-38.
T_{TX-EYE}	Transmitter Eye including all jitter sources	0.75 (min)	0.75 (min)	See Table 4-19.	UI	Does not include SSC or Refclk jitter. Includes Rj at 10^{-12}. See Notes 2, 3, 4, and 10. Note that 2.5 GT/s and 5.0 GT/s use different jitter determination methods.
$T_{TX-EYE-MEDIAN-to-MAX-JITTER}$	Maximum time between the jitter median and max deviation from the median	0.125 (max)	Not specified	Not specified	UI	Measured differentially at zero crossing points after applying the 2.5 GT/s clock recovery function. See Note 2.

Symbol	Parameter	2.5 GT/s	5.0 GT/s	8.0 GT/s	Units	Comments
$T_{TX-HF-DJ-DD}$	Tx deterministic jitter > 1.5 MHz	Not specified	0.15 (max)	See Table 4-19.	UI	Deterministic jitter only. See Notes 2 and 10.
$T_{TX-LF-RMS}$	Tx RMS jitter < 1.5 MHz	Not specified	3.0	See Table 4-19.	ps RMS	Total energy measured over a 10 kHz – 1.5 MHz range.
$T_{RF-MISMATCH}$	Tx rise/fall mismatch	Not specified	0.1 (max)	Not specified	UI	Measured from 20% to 80% differentially. See Note 2.
$RL_{TX-DIFF}$	Tx package plus Si differential return loss	10 (min)	10 (min) for 0.05 - 1.25 GHz 8 (min) for >1.25 - 2.5 GHz	10(min) for 0.05-1.25 GHz 8(min) for >1.25 - 2.5 GHz 4(min) for >2.5 - 4 GHz	dB	For details refer to Figure 4-56.
RL_{TX-CM}	Tx package plus Si common mode return loss	6 (min) for .05-2.5 GHz	6 (min) for .05-2.5 GHz	6 (min) for .05-2.5 GHz 3 (min) for 2.5 GHz	dB	For details refer to Figure 4-57.
$Z_{TX-DIFF-DC}$	DC differential Tx impedance	80 (min) 120 (max)	120 (max)	120 (max)	Ω	Low impedance defined during signaling. Parameter is captured for 5.0 GHz by $RL_{TX-DIFF}$. The (min) value is bounded by $RL_{TX-DIFF}$ for 5.0 GT/s and 8.0 GT/s.
$V_{TX-CM-AC-PP}$	Tx AC peak-peak common mode voltage (5.0 GT/s)	Not specified	150 (max)	150 (max)	mVPP	At 8.0 GT/s, no more than 50 mVPP in 0.03-500 MHz range. At 5.0 GT/s no more than 100 mVPP in 0.03 500 MHz range. See Notes 5 and 12.
$V_{TX-CM-AC-P}$	Tx AC peak common mode voltage (2.5 GT/s)	20	Not specified	Not specified	mV	See Note 5.
$I_{TX-SHORT}$	Transmitter short-circuit current limit	90 (max)	90 (max)	90 (max)	mA	The total single-ended current a transmitter can supply when shorted to ground. Note 13.
$V_{TX-DC-CM}$	Transmitter DC common-mode voltage	0 (min) 3.6 (max)	0 (min) 3.6 (max)	0 (min) 3.6 (max)	V	The allowed DC common-mode voltage at a transmitter pin under any conditions. Note 13.
$V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$	Absolute Delta of DC Common Mode Voltage during L0 and Electrical Idle.	0 (min) 100 (max)	0 (min) 100 (max)	0 (min) 100 (max)	mV	$\lvert V_{TX-CM-DC\ [during\ L0]} - V_{TX-CM-Idle-DC\ [during\ Electrical\ Idle]}\rvert$ <= 100 mV $V_{TX-CM-DC}$ = $DC_{(avg)}$ of $\lvert V_{TX-D+} + V_{TX-D-}\rvert/2$ $V_{TX-CM-Idle-DC}$= $DC_{(avg)}$ of $\lvert V_{TX-D+} + V_{TX-D-}\rvert/2$ [Electrical Idle]
$V_{TX-CM-DC-LINE-DELTA}$	Absolute Delta of DC Common Mode Voltage between D+ and D-	0 (min) 25 (max)	0 (min) 25 (max)	0 (min) 25 (max)	mV	$\lvert V_{TX-CM-DC-D+\ [during\ L0]} - V_{TX-CM-DC-D-\ [during\ L0,]}\rvert \le 25$ mV $V_{TX-CM-DC-D+}$ = $DC_{(avg)}$ of $\lvert V_{TX-D+}\rvert$ [during L0] $V_{TX-CM-DC-D-}$ = $DC_{(avg)}$ of $\lvert V_{TX-D-}\rvert$ [during L0]

Symbol	Parameter	2.5 GT/s	5.0 GT/s	8.0 GT/s	Units	Comments
$V_{TX-IDLE-DIFF-AC-p}$	Electrical Idle Differential Peak Output Voltage	0 (min) 20 (max)	0 (min) 20 (max)	0 (min) 20 (max)	mV	$V_{TX-IDLE-DIFF-AC-p} = \lvert V_{TX-Idle-D+} - V_{Tx-Idle-D-} \rvert \leq 20$ mV. Voltage must be band pass filtered to remove any DC component and HF noise. The bandpass is constructed from two first-order filters, the high pass and low pass 3 dB bandwidths are 10 kHz and 1.25 GHz, respectively.
$V_{TX-IDLE-DIFF-DC}$	DC Electrical Idle Differential Output Voltage	Not specified	0 (min) 5 (max)	0 (min) 5 (max)	mV	$V_{TX-IDLE-DIFF-DC} = \lvert V_{TX-Idle-D+} - V_{Tx-Idle-D-} \rvert \leq 5$ mV. Voltage must be low pass filtered to remove any AC component. The low pass filter is first-order with a 3 dB bandwidth of 10 kHz.
$V_{TX-RCV-DETECT}$	The amount of voltage change allowed during Receiver Detection	600 (max)	600 (max)	600 (max)	mV	The total amount of voltage change in a positive direction that a Transmitter can apply to sense whether a low impedance Receiver is present. Note: Receivers display substantially different impedance for $V_{IN} < 0$ vs $V_{IN} > 0$. See Table 4-24 for details.
$T_{TX-IDLE-MIN}$	Minimum time spent in Electrical Idle	20 (min)	20 (min)	20 (min)	ns	Minimum time a Transmitter must be in Electrical Idle.
$T_{TX-IDLE-SET-TO-IDLE}$	Maximum time to transition to a valid Electrical Idle after sending an EIOS	8 (max)	8 (max)	8 (max)	ns	After sending the required number of EIOSs, the Transmitter must meet all Electrical Idle specifications within this time. This is measured from the end of the last UI of the last EIOS to the Transmitter in Electrical Idle.
$T_{TX-IDLE-TO-DIFF-DATA}$	Maximum time to transition to valid diff signaling after leaving Electrical Idle	8 (max)	8 (max)	8 (max)	ns	Maximum time to transition to valid diff signaling after leaving Electrical Idle. This is considered a debounce time to the Tx.
$T_{CROSSLINK}$	Crosslink random timeout	1.0 (max)	1.0 (max)	1.0 (max)	ms	This random timeout helps resolve potential conflicts in the crosslink configuration.
$L_{TX-SKEW}$	Lane-to-Lane Output Skew	500 ps + 2 UI (max)	500 ps + 4 UI (max)	500 ps + 6 UI	ps	Between any two Lanes within a single Transmitter.
C_{TX}	AC Coupling Capacitor	75 (min) 265 (max)	75 (min) 265 (max)	176 (min) 265 (max)	nF	All Transmitters shall be AC coupled. The AC coupling is required either within the media or within the transmitting component itself. Note 14.

Notes:

1. SSC permits a +0, - 5000 ppm modulation of the clock frequency at a modulation rate not to exceed 33 kHz.

2. Measurements at 5.0 GT/s require an oscilloscope with a bandwidth of ≥12.5 GHz, or equivalent, while measurements made at 2.5 GT/s require a scope with at least 6.2 GHz bandwidth. Measurement at 5.0 GT/s must deconvolve effects of compliance test board to yield an effective measurement at Tx pins. 2.5 GT/s may be measured within 200 mils of Tx device's pins, although deconvolution is recommended. For measurement setup details, refer Figure 4-32 to and Figure 4-33. At least 10^6 UI of data must be acquired.
3. Transmitter jitter is measured by driving the Transmitter under test with a low jitter "ideal" clock and connecting the DUT to a reference load.
4. Transmitter raw jitter data must be convolved with a filtering function that represents the worst case CDR tracking BW. 2.5 GT/s and 5.0 GT/s use different filter functions that are defined in Figure 4-49. After the convolution process has been applied, the center of the resulting eye must be determined and used as a reference point for obtaining eye voltage and margins.
5. $V_{TX\text{-}AC\text{-}CM\text{-}PP}$ and $V_{TX\text{-}AC\text{-}CM\text{-}P}$ are defined in Section 4.3.3.2. Measurement is made over at least 10^6 UI.
6. The Tx PLL Bandwidth must lie between the min and max ranges given in the above table. PLL peaking must lie below the value listed above. Note: the PLL B/W extends from zero up to the value(s) specified in the above table.
7. Measurements are made for both common mode and differential return loss. The DUT must be powered up and DC isolated, and its data+/data- outputs must be in the low-Z state at a static value.
8. A single combination of PLL BW and peaking is specified for 2.5 GT/s implementations. For 5.0 GT/s, two combinations of PLL BW and peaking are specified to permit designers to make a tradeoff between the two parameters. If the PLL's min BW is ≥8 MHz, then up to 3.0 dB of peaking is permitted. If the PLL's min BW is relaxed to ≥5.0 MHz, then a tighter peaking value of 1.0 dB must be met. In both cases, the max PLL BW is 16 MHz.
9. Reduced swing output, defined by $V_{TX\text{-}DIFF\text{-}PP\text{-}LOW}$ must be implemented as shown in Figure 4-37 with no de-emphasis.
10. For 5.0 GT/s, de-emphasis timing jitter must be removed. An additional HPF function must be applied as shown in Figure 4-49. This parameter is measured by accumulating a record length of 10^6 UI while the DUT outputs a compliance pattern. $T_{MIN\text{-}PULSE}$ is defined to be nominally 1 UI wide and is bordered on both sides by pulses of the opposite polarity. Refer to Figure 4-38.
11. Root Complex Tx de-emphasis is configured from Upstream controller. Downstream Tx de-emphasis is set via a command, issued at 2.5 GT/s. For details, refer to the appropriate location in Section 4.2.
12. Tx CM noise for 8.0 GT/s is measured at TP1 without de-embedding the breakout channel. The parameter captures device CM noise only and is not intended to capture system CM noise. For 5.0 GT/s an LPF with a -3 dB corner at 2.5 GHz is applied to the raw data. For 8.0 GT/s the filter's -3 dB corner is at 4.0 GHz.
13. $I_{TX\text{-}SHORT}$ and $V_{TX\text{-}DC\text{-}CM}$ stipulate the maximum current/voltage levels that a transmitter can generate and therefore define the worst case transients that a receiver must tolerate.
14. All platforms that have transmitters supporting 8.0 GT/s must implement the 176-265 nF C_{TX} value. Platforms operating at 2.5 or 5.0 GT/s only may implement over a range of 75 to 265 nF.

4.3.3.13.1. 8.0 GT/s Specific Tx Voltage and Jitter Parameter Table

Table 4-19: 8.0 GT/s Specific Tx Voltage and Jitter Parameters

Symbol	Parameter	Value	Units	Notes
$V_{TX\text{-}FS\text{-}NO\text{-}EQ}$	Full swing Tx voltage with no TxEq	1300 (max) 800 (min)	mVPP	Note 1
$V_{TX\text{-}RS\text{-}NO\text{-}EQ}$	Reduced swing Tx voltage with no TxEq	1300 (max)	mVPP	Note 1
$V_{TX\text{-}EIEOS\text{-}FS}$	Min swing during EIEOS for full swing	250 (min)	mVPP	Note 2

Symbol	Parameter	Value	Units	Notes
$V_{TX-EIEOS-RS}$	Min swing during EIEOS for reduced swing	232 (min)	mVPP	Note 2
T_{TX-UTJ}	Tx uncorrelated total jitter	31.25 (max)	ps PP @ 10^{-12}	
$T_{TX-UDJDD}$	Tx uncorrelated deterministic jitter	12 (max)	ps PP	
$T_{TX-UPW-TJ}$	Total uncorrelated PWJ	24 (max)	ps PP @ 10^{-12}	Notes 3,4
$T_{TX-UPW-DJDD}$	Deterministic DjDD uncorrelated PWJ	10 (max)	ps PP	Note 3,4
T_{TX-DDJ}	Data dependent jitter	18 (max)	ps PP	Notes 4,5
$ps21_{TX}$	Pseudo package loss	-3.0 (min)	dB	PP ratio of 64 ones/64 zeroes pattern vs. 0101 pattern. No Tx equalization. Note 6.
$V_{TX-BOOST-FS}$	Tx boost ratio for full swing	8.0 (min)	dB	Assumes ±1.5 dB tolerance from diagonal elements in Figure 4-45.
$V_{TX-BOOST-RS}$	Tx boost ratio for reduced swing	2.5 (min)	dB	Assumes ±1.0 dB tolerance from diagonal elements in Figure 4-45.
$EQ_{TX-COEFF-RES}$	Tx coefficient resolution	1/24 (max) 1/63 (min)	N/A	

Notes:

1. Voltage measurements for $V_{TX-FS-NO-EQ}$ and $V_{TX-RS-NO-EQ}$ are made using the 64-zeroes/64-ones pattern in the compliance pattern
2. Voltage limits comprehend both full swing and reduced swing modes. The Tx must reject any changes that would violate this specification. The maximum level is covered in the $V_{TX-FS-NO-EQ}$ measurement which represents the maximum peak voltage the Tx can drive. The $V_{TX-EIEOS-FS}$ and $V_{TX-EIEOS-RS}$ voltage limits are imposed to guarantee the EIEOS threshold of 175 mVPP at the Rx pin. This parameter is measured using the actual EIEOS pattern that is part of the compliance pattern and then removing the ISI contribution of the breakout channel. The transmitter must advertise a value for LF during TS1 at 8.0 GT/s that ensures that these parameters are met.
3. PWJ parameters shall be measured after DDJ separation.
4. Measured with optimized preset value after de-embedding to Tx pin.
5. The 18 ps number takes into account measurement error.
6. The -3.0 dB number takes into account measurement error. For some Tx package/driver combinations $ps21_{TX}$ may be greater than 0 dB.

4.3.4. Receiver Specifications

Both 5.0 GT/s and 8.0 GT/s Receivers are tested by means of a stressed eye technique, where a signal with specific impairments is applied to the Rx DUT, and the BER is monitored. Due to substantial differences between the 2.5/5.0 GT/s and 8.0 GT/s stressed eye methods, the two are presented in separate subsections. A third subsection defines parameters that are common across all speeds.

4.3.4.1. 2.5 GT/s and 5.0 GT/s Receiver Specification

For 2.5 GT/s, the parameters defined in Table 4-24 are defined at the Receiver pins. As such, they do not directly measure a Receiver's performance. There is instead an implied correlation between the margins observed at the Receiver's pins and the BER, but no prescribed methodology for measuring it.

At 5.0 GT/s it becomes necessary to define a performance-based methodology for tolerancing a Receiver. Tolerancing is a two step procedure in which a test apparatus is calibrated to yield the worst case signal margins defined in Table 4-20 or Table 4-21 into a test load. The margins applied to a Receiver are dependent on the clocking architecture with which the Receiver will operate. Therefore two different sets of margining parameters are defined for the common Refclk Rx and data clocked Rx architectures. The Receiver under test then replaces the test load, and its BER is observed. The following sections describe the Receiver tolerancing procedure in detail.

4.3.4.1.1. Receiver Pin to Pad Correlation

For 2.5 GT/s signaling, sufficient accuracy is possible by making *in situ* measurements at a Receiver's pins, notwithstanding that such measurements include both incident and reflected signals, and that the Receiver plus its package is not an ideal termination load.

At 5.0 GT/s a more accurate measurement method is required, and this is achieved by use of a two step Receiver tolerancing procedure. A signal source driving a standard 2 x 50 Ω test load is adjusted to yield worst case margins as defined in Table 4-20 or Table 4-21. Then the test load is replaced by the Receiver under test, and its BER is observed.

In developing the Rx specification margins, allowance is made to account for signal degradation between what is measured at the reference load and what is required at the Receiver device's pads. It is the shared responsibility of the Receiver silicon and the package designers to comprehend package-silicon interactions and guarantee that the signal levels appearing at the device's pads will meet the 10^{-12} BER requirement.

4.3.4.1.2. Calibration Channel Characteristics

A calibration channel provides an easily implemented means of generating data with a substantial variation in UI to UI voltage swing. Such a variation is important to stress a Receiver and is comprehended via the $V_{RX\text{-}MIN\text{-}MAX\text{-}RATIO}$ parameter in Table 4-20 and Table 4-21. The use of a calibration channel permits this parameter to be stressed without requiring that the pattern generator implement de-emphasis or other UI to UI voltage variation.

The calibration channel is characterized in terms of its transient response and return loss. Analysis has shown that, for a binary, non-equalized input, a channel requires a length of approximately 28 inches when fabricated on conventional FR4 PCB material. The time domain characteristics of the calibration channel may be verified by driving it with a non-equalized signal consisting of worst case 8b/10b pattern and observing a $T_{RX\text{-}MIN\text{-}PULSE}$ of approximately 150 ps when no differential Dj voltage is applied.

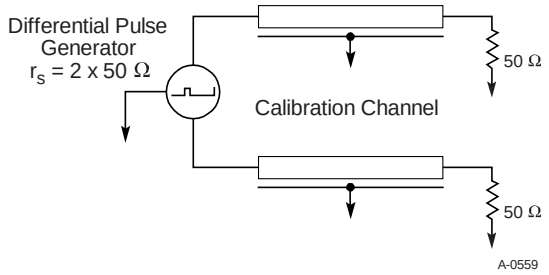


Figure 4-58: Calibration Channel Validation

Figure 4-59 illustrates signals as seen at the driver and Receiver. Note that the Tx data is non-equalized.

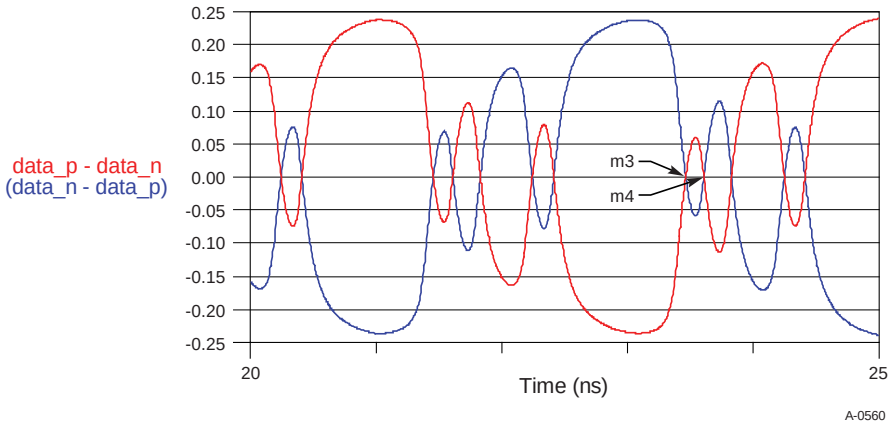


Figure 4-59: Calibration Channel Showing $T_{MIN\text{-}PULSE}$

In addition to its forward transient domain characteristics, the calibration channel must also be defined in terms of its return loss, as shown in Figure 4-60, where the RL limit is 18 dB. This measurement may be made as a single-ended Port measurement with the far end of the channel terminated to ground. Note that the calibration channel may be fabricated as two widely spaced traces yielding single-ended impedance that is 0.5x the differential impedance. Such a topology eliminates the need for performing both single-ended and differential return loss measurements.

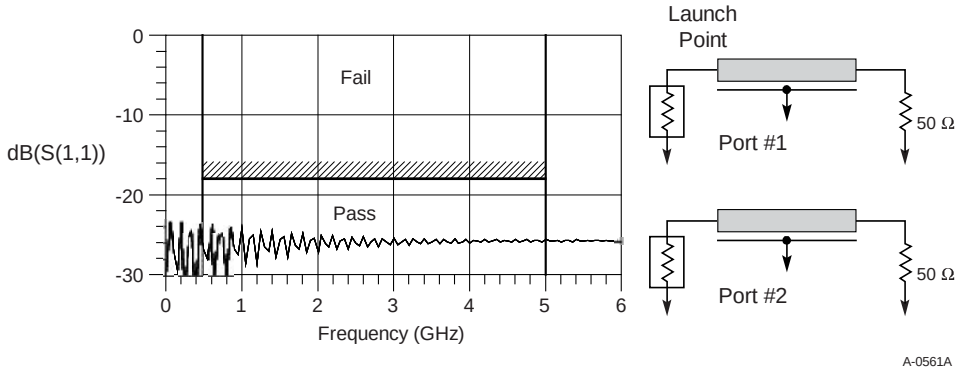


Figure 4-60: Calibration Channel $|s_{11}|$ Plot with Tolerance Limits

4.3.4.1.3. Receiver Tolerancing at 5.0 GT/s

For 5.0 GT/s, a clearly delineated methodology for tolerance testing the Receiver is defined to permit an easily implemented test setup. Figure 4-61 illustrates a functional block diagram of the Receiver tolerancing setup. The voltage sources on the left side supply signals representing various jitter sources that a Receiver may see. A compliance data pattern is generated by a pattern generator which may also furnish a subrate 100 MHz clock. Common mode and differential mode voltage sources are added to the pattern generator's output via power splitters. Finally, ISI effects are generated by a calibration channel, which will be described in detail in a later section.

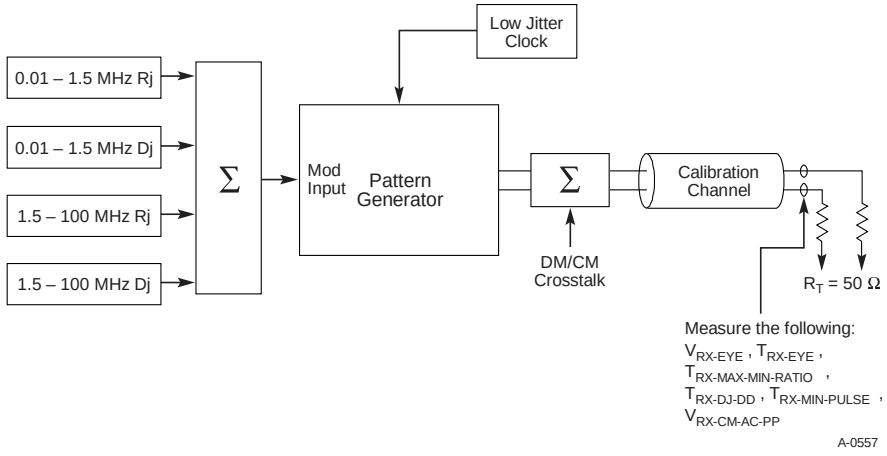


Figure 4-61: Setup for Calibrating Receiver Test Circuit into a Reference Load

Two separate tolerancing tables are defined to accommodate the differing jitter limits that occur for common Refclk Rx and data clocked Rx architectures. A Receiver typically implements only one of the two clock architectures, and so only needs to be tested against the relevant one.

Implementation and calibration details for the Receiver test setup apparatus are typically equipment specific and lie outside the scope of this specification.

Once the test setup has been calibrated the test load is replaced by the Receiver under test as shown in Figure 4-62. The test must be run for a sufficient time to permit a statistically meaningful number of UI to be received so that a BER figure can be obtained.

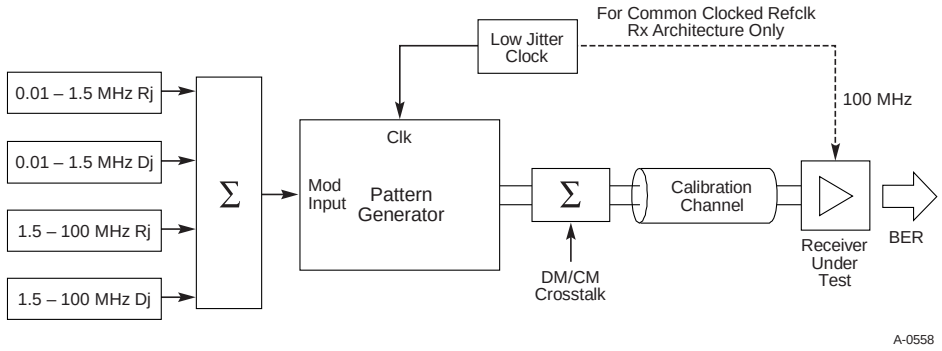


Figure 4-62: Setup for Testing Receiver

Table 4-20: 5.0 GT/s Tolerancing Limits for Common Refclk Rx Architecture

Parameter	Description	Min	Max	Units	Notes
UI	Unit interval without including of SSC	199.94	200.06	ps	Over 10^6 UI
$T_{RX-HF-RMS}$	1.5 – 100 MHz RMS jitter		3.4	ps RMS	Spectrally flat, 3
$T_{RX-HF-DJ-DD}$	Max Dj impinging on Rx under test		88	ps	2,4
$T_{RX-SSC-RES}$	33 kHz Refclk residual	--	75	ps	
$T_{RX-LF-RMS}$	< 1.5 MHz RMS jitter	--	4.2	ps RMS	Spectrally flat
$T_{RX-MIN-PULSE}$	Minimum single pulse applied at Rx	120		ps	2
$V_{RX-MIN-MAX-RATIO}$	Min/max pulse voltage ratio seen over an time interval of 2 UI	--	5		2
V_{RX-EYE}	Receive eye voltage opening	120		mVPP diff	1,3
$V_{RX-CM-CH-SRC}$	Common mode noise from Rx	--	300	mVPP	2

Notes:

1. Refer to Figure 4-63 for a description of how the Rx eye voltage is defined.
2. Accumulated over 10^6 UI.
3. Minimum eye is obtained by first injecting maximum Dj and then adjusting Rj until a minimum eye (defined by T_{RX-EYE} as shown in Figure 4-63) is reached. Rj is spectrally flat before being filtered with a BPF having 3 dB cut-offs f_{C-LOW} and f_{C-HIGH} of 1.5 MHz and 100 MHz, respectively with step rolloff at 1.5 MHz and a 20 dB/decade rolloff on the high side. Minimum eye width is defined for a sample size equivalent to a BER of 10^{-12}.
4. Different combinations of $T_{RX-HF-DJ-DD}$ and $T_{RX-HF-RMS}$ are needed to measure $T_{RX-TJ-CC}$ and $T_{RX-DJ-DD-CC}$.

Table 4-21: 5.0 GT/s Tolerancing Limits for Data Clocked Rx Architecture

Parameter	Description	Min	Max	Units	Notes
UI	Unit interval without including of SSC	199.94	200.06	ps	Over 10^6 UI
$T_{RX-HF-RMS}$	1.5 – 100 MHz RMS jitter		4.2	ps RMS	Spectrally flat, 3
$T_{RX-HF-DJ-DD}$	Max Dj impinging on Rx under tolerancing		88	ps	2,4
$T_{RX-LF-SSC-FULL}$	Full 33 kHz SSC	--	20	ns	2
$T_{RX-LF-RMS}$	10 kHz to 1.5 MHz RMS jitter	--	8.0	ps RMS	Spectrally flat
$T_{RX-MIN-PULSE}$	Minimum single pulse applied at Rx	120		ps	2
$V_{RX-MIN-MAX-RATIO}$	Min/max pulse voltage ratio seen over an time interval of 2 UI	--	5		2
V_{RX-EYE}	Receive eye voltage opening	100		mVPP diff	1,3
$V_{RX-CM-CH-SRC}$	Common mode noise from Rx	--	300	mVPP	2

Notes:

1. Refer to Figure 4-63 for a description of how the Rx eye voltage is defined.
2. Accumulated for 10^6 UI.
3. Minimum eye is obtained by first injecting maximum Dj and then adjusting Rj until a minimum eye (defined by T_{RX-EYE} as shown in Figure 4-63) is reached. Rj is spectrally flat before being filtered with a BPF having 3 dB cut-offs f_{C-LOW} and f_{C-HIGH} of 1.5 MHz and 100 MHz, respectively with step rolloff at 1.5 MHz and a 20 dB/decade rolloff on the high side. Minimum eye width is defined for a sample size equivalent to a BER of 10^{-12}.
4. Different combinations of $T_{RX-HF-DJ-DD}$ and $T_{RX-HF-RMS}$ are needed to measure $T_{RX-TJ-DC}$ and $T_{RX-DJ-DD-DC}$.

4.3.4.1.4. 5.0 GT/s Receiver Compliance Eye Diagram

Voltage and time margins for the Receiver are defined via an eye diagram, as illustrated in Figure 4-63. The margins are defined as they must appear at the test load at the far end of the channel. Note: The center of the eye is determined by the same phase jitter filter function used by the Transmitter.

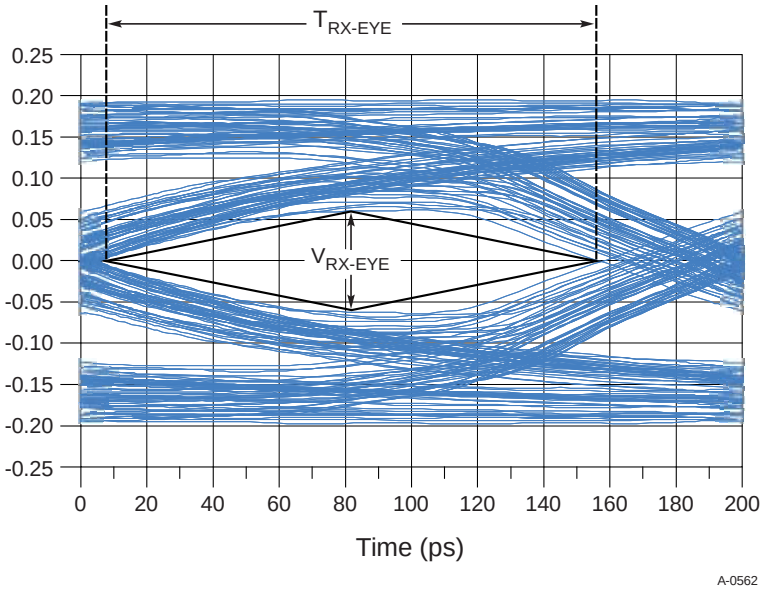


Figure 4-63: Receiver Eye Margins

4.3.4.1.5. 5.0 GT/s Receiver Dynamic Voltage Range

Receiver dynamic range is defined by two parameters: $V_{RX\text{-}EYE}$ and $V_{RX\text{-}MAX\text{-}MIN\text{-}RATIO}$. The $V_{RX\text{-}EYE}$ parameter defines the range over which any Receiver must operate at any time, and in any system. This parameter's lower and upper limits are typically constrained by Rx sensitivity and Receiver's ESD and bias limitations, respectively. The second parameter, $V_{RX\text{-}MAX\text{-}MIN\text{-}RATIO}$, defines the voltage range ratio over which a particular Receiver must operate for consecutive UI. Figure 4-64 shows a typical voltage plot into a reference load that yields a near worst case $V_{RX\text{-}MAX\text{-}MIN\text{-}RATIO}$. $V_{SWING\text{-}MAX}$ is defined in relation to $V_{SWING\text{-}MIN}$ over an interval of 2.0 UI, as illustrated below. Typically, the right hand side of the 2 UI interval is placed on the peak of the waveform corresponding to $V_{SWING\text{-}MIN}$. The 2 UI separation guarantees that $V_{SWING\text{-}MAX}$. A 2 UI interval guarantees that $V_{SWING\text{-}MAX}$ is measured on the flat portion of its curve and accounts for worst case jitter and dispersive channel effects.

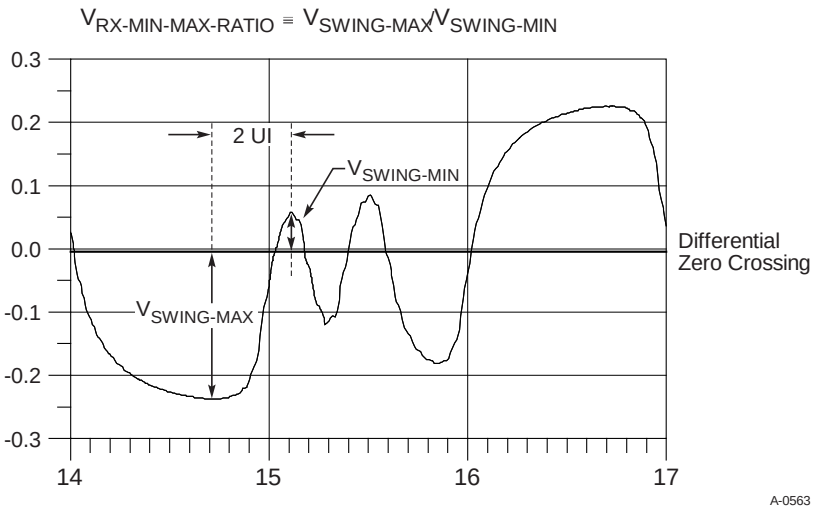


Figure 4-64: Signal at Receiver Reference Load Showing Min/Max Swing

4.3.4.2. 8.0 GT/s Receiver Stressed Eye Specification

An 8.0 GT/s receiver is tested by means of applying a stressed eye to the DUT and verifying that the receiver meets the 10^{-12} BER target. The method for 8.0 GT/s Rx testing is conceptually similar to that defined for 5.0 GT/s, although there are some implementation differences, primarily due to the higher bit rate and the measurement challenges it imposes.

4.3.4.3. Breakout and Replica Channels

As with the 8.0 GT/s transmitter, the closest practical measurement points to the Rx DUT are the coaxial connectors at the end of a breakout channel. By constructing a replica channel that closely matches the electrical characteristics of the breakout channel it is possible to measure the signal as it would appear at the DUT's pin if the DUT were an ideal termination. The impedance targets for the interconnect environment are 100 Ω differential and 50 Ω single-ended.

In Figure 4-65 the stressed eye is observed at TP2 with the signal sources connected to the calibration channel. Once the stressed eye has been calibrated, the signal source is applied to the DUT. Note that TP1-TP2 encompasses all the components between the signal source and the equivalent of the DUT pin, thereby capturing all non-ideal characteristics in the overall insertion loss due to cabling, etc. The AC and DC loss from generator to TP1 are assumed to be zero or must be otherwise de-embedded.

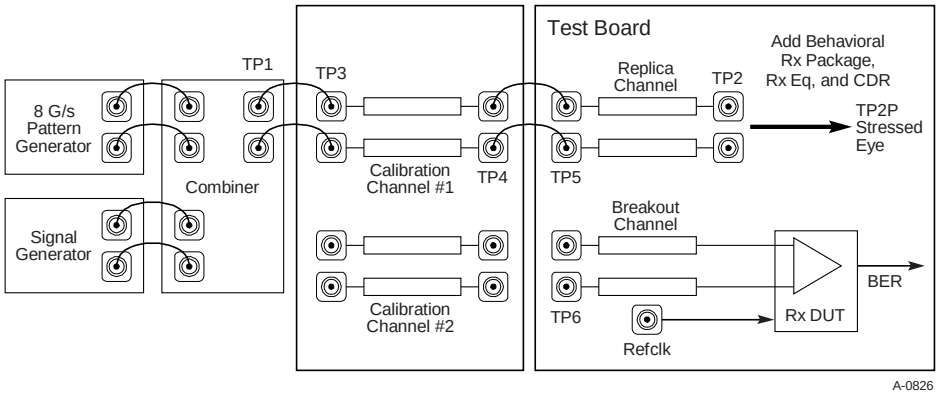


Figure 4-65: Rx Testboard Topology

4.3.4.3.1. Calibration Channels

Calibration channels with differing insertion loss provide a convenient means of generating prescribed amounts of ISI. Three channel loss characteristics provide a means of requiring the Rx DUT to adjust its equalizer over a range representative of the minimum to maximum length channels encountered in real platforms. The impedance targets for the Rx tolerancing interconnect environment are 100 Ω differential and 50 Ω single-ended. The 8.0 GT/s calibration channel must meet the 18 dB return loss limit defined in Figure 4-60 over a frequency range of 50 MHz to 8 GHz.

A calibration channel consists of a differential pair of PCB traces terminated at both ends by coaxial connectors. The calibration channel's electrical characteristics are defined in terms of differential insertion loss masks as shown in Figure 4-66, with S_{21} measured between TP1 and TP2. Any HF loss caused by the generator and combiner and the connections between TP1-TP3 and TP4-TP5 that represent cabling are to be included in the S_{21} measurement.

While the s-parameter mask does not extend below 1.0 GHz, the calibration channels must be well behaved below 1.0 GHz and must not have a DC resistance in excess of 7.5 ohms, as measured by the sum of the resistances of the D+ and D- traces. This limitation on DC resistance guarantees that the calibration channel low frequency characteristic is consistent with the extrapolations to DC of the S_{dd21} masks shown in Figure 4-66.

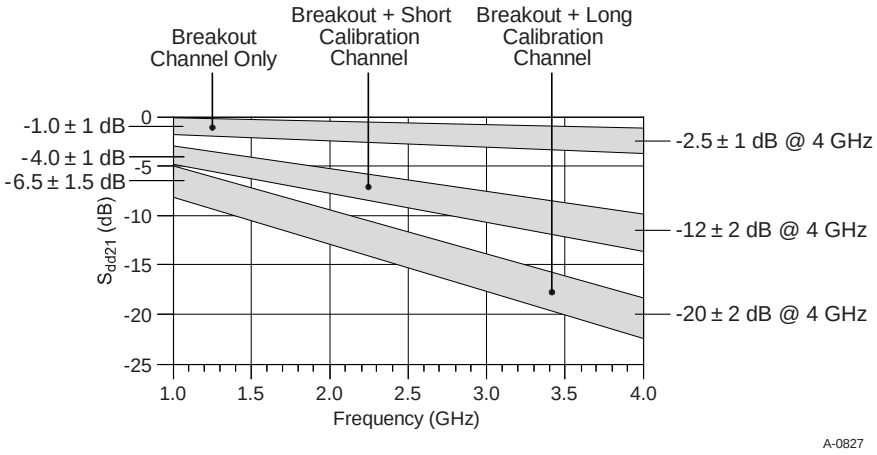


Figure 4-66: Insertion Loss Guidelines for Calibration/Breakout Channels

4.3.4.3.2. Post Processing Procedures

In order to establish a 1:1 correlation between the receiver and channel specifications it is necessary to define a common reference point between the two. This point is defined as the abstracted input to the Rx latch after Rx equalization has been applied and is designated as TP2P on Figure 4-71 and Figure 4-73. For the receiver test this requires that the stressed eye characteristics must be observed at a point different from where the signal is actually measured and imposes the need for post processing steps outlined in the following sections.

4.3.4.3.3. Behavioral Rx Package Model

A behavioral Rx package model is included as part of the post processing to allow the calibrated eye to include package insertion loss. Details of the behavioral Rx package may be found in Section 4.3.6.4.1 of the Channel Tolerancing section. An s-parameter model for the behavioral Rx package model is available as design collateral. Note that the termination at the pad side of the package model is assumed to be 2 x 50 Ω.

4.3.4.3.4. Behavioral CDR Model

Post processing shall include a behavioral CDR model with a transfer function defined in Figure 4-67. The CDR algorithm must generate an eye diagram based on the signal referenced, to an extracted and regenerated data clock.

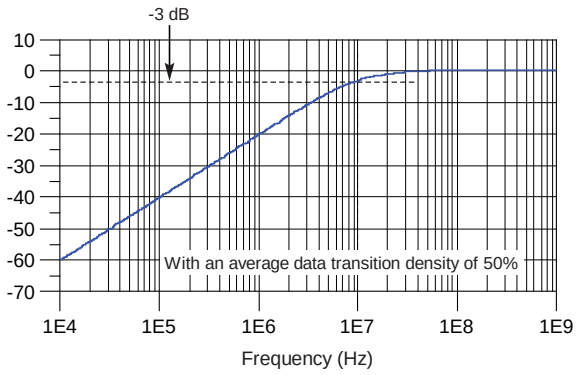


Figure 4-67: Behavioral CDR Model for Rx Measurement

4.3.4.3.5. Behavioral Rx Equalization Algorithms

For the longest calibration channel the stressed eye will be closed, making direct measurement of the stressed eye jitter parameters unfeasible. This problem is overcome by employing a behavioral receiver equalizer that implements both a 1st order CTLE and a 1-tap DFE. For the short and medium calibration channels the behavioral Rx equalizer shall implement a 1st order CTLE only.

Rx behavioral CTLE and/or DFE are intended only a means of obtaining an open eye in the presence of calibration channel ISI plus the other signal impairment terms. The behavioral Rx equalization algorithm is not intended to serve as a guideline for implementing actual receiver equalization.

4.3.4.3.6. Behavioral CTLE

8.0 GT/s behavioral Rx equalization defines a 1st order CTLE with fixed LF and HF poles, and an adjustable DC gain (A_{DC}) specified according to the family of curves shown in Figure 4-69. A_{DC} is adjustable over a minimum range of -6 to -12 dB in steps of 1.0 dB.

$$H(s) = \omega_{P2} \frac{s + \omega_{P1} * A_{DC}}{(s + \omega_{P1}) * (s + \omega_{P2})}$$

ω_{P1} = pole 1 = 2π*2 GHz
ω_{P2} = pole 2 = 2π*8 GHz
A_{DC} = dc gain

A-0828

Figure 4-68: Transfer Function for Behavioral CTLE

The following diagram illustrates the gain vs. frequency behavior of the CTLE as A_{DC} is varied over its minimum to maximum range in 1.0 dB steps.

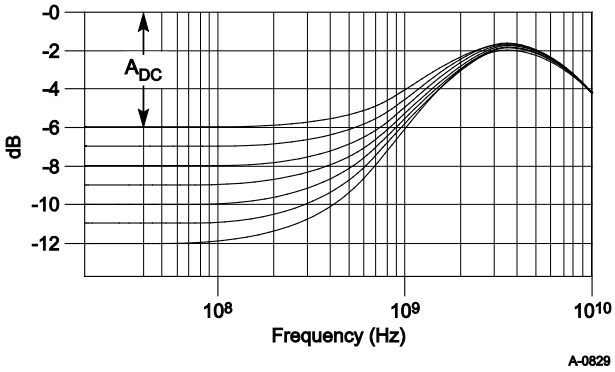
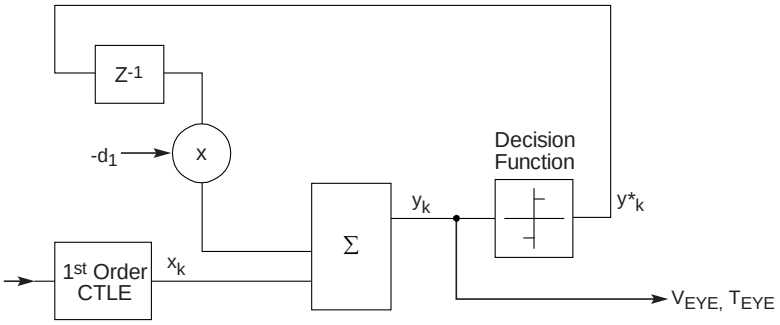


Figure 4-69: Loss Curves for Behavioral CTLE

4.3.4.3.7. Behavioral DFE

In addition to the 1st order CTLE, a one-tap DFE algorithm is required for calibrating the stressed eye when employing the max length calibration channel. The DFE may be represented by the following equation and flow diagram. The limits on d_1 are ±30 mV.



$y_k = x_k - d_1 \mathrm{sgn}(y_{k-1})$

y_k = DFE summer differential output voltage.

y^*_k = decision function output voltage. $|y^*_k| = 1$

x_k = DFE differential input voltage

d_1 = feedback coefficient

k = sample index in UI

A-0830

Figure 4-70: Equation and Flow Diagram for 1-tap DFE

4.3.4.4. Stressed Voltage and Stressed Jitter Rx Tests

Two distinct tests are utilized to test a receiver: one for its minimum eye height (voltage), and another for its minimum eye width (jitter). The procedures for calibrating the stressed eye are similar, although the number and magnitude of signal impairment sources varies between the two tests.

When testing a receiver it is required to have other PCI Express Lanes on the DUT sending or receiving data. Similarly, if the device supports other I/O, it should also be sending or receiving on these interfaces. The goal is to have the Rx test environment replicate that found in a real system as closely as possible.

4.3.4.4.1. Procedure for Calibrating a Stressed Voltage Eye

The configuration for calibrating a stressed voltage eye for Rx testing is shown in Figure 4-71 where the calibration procedure is performed for all three calibration/breakout channel combinations, as shown in Figure 4-66. Generator de-emphasis for the -20 dB and -2.5 dB channels is 0.0 dB, while the -12 dB channel requires 3.5 dB of pre-shoot and -6 dB of de-emphasis. Up to ±2.0 dB of additional generator de-emphasis may be applied to generate an effective S_{21} to meet the insertion loss masks illustrated in Figure 4-66.

Rj and Sj are added as defined in Table 4-22, and common mode and differential mode noise sources are added simultaneously.

All stressed eye tests use the modified Tx compliance pattern as defined in Section 4.2.11 of the protocol section. This pattern consists of 65 K blocks, each 130 bits in length, consisting mostly of PRBS23.

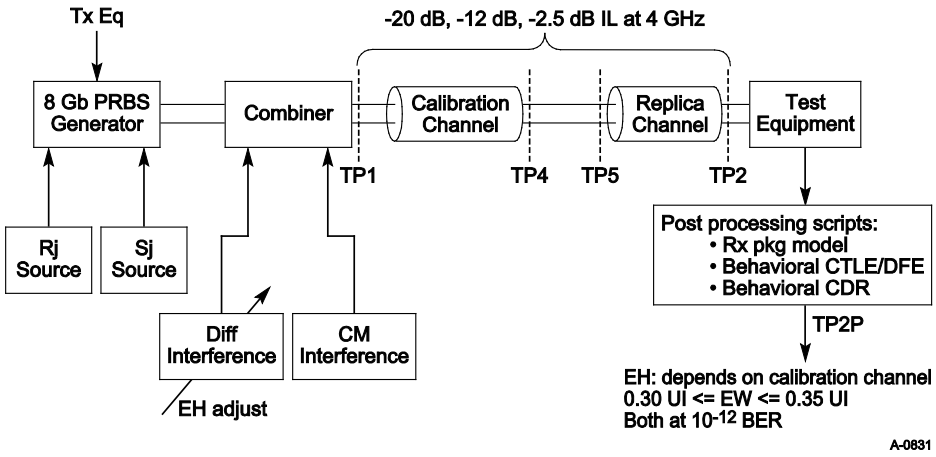


Figure 4-71: Setup for Calibrating the Stressed Voltage Eye

Eye width and eye height are defined after applying post processing and are defined at TP2P. TP1 represents the output of the combiner or generator (if the combiner is internal to the generator). TP2 represents the signal at the far end of the breakout channel at Rx DUT's pin, as if the DUT were an ideal termination. The long calibration channel utilizes both CTLE and DFE, while the medium and short channels calibration channels use CTLE only. EH is set by adjusting the amount of differential noise until the value defined by $V_{RX\text{-}SV\text{-}8G}$ is obtained. If it is not possible to maintain a sufficient eye width by adjusting only the differential noise, it is acceptable to inject less differential noise and adjust the generator launch voltage.

Calibration of a stressed eye requires that the Rx equalization be optimized to yield the largest eye, where this metric is defined by the product of peak eye height times eye width (see Figure 4-88). Note that the peak EH may not occur at the UI center, but EH is defined only within an aperture of ±0.1 UI from the median center of the UI. See Figure 4-88 for details. In other words, EH values outside of this aperture are not considered when determining max EH. During the calibration process the Tx equalization remains fixed. Optimization of the Rx equalization shall be implemented

using a search algorithm that is equivalent to the algorithm described for channel compliance in Section 4.3.6.5.3. The stressed eye needs to be measured at the equivalent of TP2P after applying the effects of behavioral package, CDR, CTLE and DFE. After applying all of the eye stresses the signal at TP2 has small amplitudes and a wide dynamic range making a high accuracy measurement difficult with available test equipment. To improve the accuracy of this measurement it may be necessary to measure the various stresses applied to the generator signal independently at either TP1 or TP2 and to then combine these mathematically to calculate the final stressed eye opening at a probability level of 10^{-12}.

Table 4-22: Stressed Voltage Eye Parameters

Symbol	Parameter	Limits at 8.0 GT/s	Units	Comments
$V_{RX-LAUNCH-8G}$	Generator launch voltage	800	mVPP	Measured at TP1 Figure 4-65. $V_{RX-LAUNCH-8G}$ may be adjusted if necessary to yield the proper EH as long as the outside eye voltage at TP2 does not exceed 1300 mVPP.
$T_{RX-UI-8G}$	Unit Interval	125.00	ps	Nominal value is sufficient for Rx tolerancing. Value does not account for SSC.
$V_{RX-SV-8G}$	Eye height at TP2P	25 (-20 dB channel) 50 (-12 dB channel) 200 (-3 dB channel)	mVPP	Eye height @ BER=10^{-12}. Notes 1,2.
$T_{RX-SV-8G}$	Eye width at TP2P	0.3 to 0.35	UI	Eye width at BER=10^{-12}. Note 2
$V_{RX-SV-DIFF-8G}$	Differential mode interference	14 or greater	mVPP	Adjusted to set EH. Frequency = 2.10 GHz. Note 3.
$V_{RX-SV-CM-8G}$	Rx AC Common mode voltage at TP2P	150 (EH < 100 mVPP) 250 (EH ≥ 100 mVPP)	mVPP	Defined for a single tone at 120 MHz. Note 3.
$T_{RX-SV-SJ-8G}$	Sinusoidal Jitter at 100 MHz	0.1	UI PP	Fixed at 100 MHz. Note 4.
$T_{RX-SV--RJ-8G}$	Random Jitter	2.0	ps RMS	Rj spectrally flat before filtering. Notes 4,5.
$V_{RX-MAX-SE-SW}$	Max single-ended swing	±300	mVP	Note 6.

Notes:

1. $V_{RX-SV-8G}$ is tested at three different voltages to ensure the Rx DUT is capable of equalizing over a range of channel loss profiles. The test also guarantees the Rx is capable of operating over a sufficient dynamic range of eye heights. The "SV" in the parameter names refers to stressed voltage.
2. $V_{RX-ST-8G}$ and $T_{RX-ST-8G}$ are referenced to TP2P and are obtained after post processing data captured at TP2. $V_{RX-ST-8G}$ and $T_{RX-ST-8G}$ include the effects of applying the behavioral Rx model and Rx behavioral equalization.
3. $V_{RX-SV-DIFF-8G}$ measurement is made at TP2 without post processing. $V_{RX-SV-CM-8G}$ may be made at either TP1 or TP2. $V_{RX-SV-DIFF-8G}$ voltage may need to be adjusted over a wide range for the different loss calibration channels.
4. $T_{RX-SV-SJ-8G}$ and $T_{RX-SV-RJ-8G}$ measurements are made at TP1 without post processing.
5. Rj is applied over the following range. The low frequency limit may be between 1.5 and 10 MHz, and the upper limit is 1.0 GHz. See Figure 4-74 for details.

6. $V_{RX-MAX-SE-SW}$ sets the maximum outer, single-ended eye voltage limit in the presence of differential and CM noise applied to the Rx, as observed at TP2 relative to ground with no behavioral RxEq post processing.

4.3.4.4.2. Procedure for Testing Rx DUT for Stressed Voltage

Once the calibrated EH and EW are obtained, the cables on the test-board are reconnected such that the calibration channel drives the DUT through the breakout channel as shown in Figure 4-72. At this time the generator's equalization may be adjusted to yield an optimum combination of Tx/Rx equalization. The means by which the generator's and the DUT's equalization settings are optimized are not defined in the base specification.

A generator's equalization capability may match, but must not exceed, the Tx equalization capability defined in the Tx specification. Specifically, the generator equalizer may support one pre-cursor and one post-cursor FIR tap with worst case resolution and dynamic range as defined in Table 4-19, and the minimum generator FIR resolution must be 1/24. The means by which the generator's and the DUT's equalization settings are optimized is not defined in the base specification. The stressed voltage eye test is repeated for each of the three calibration channels, and the Tx/Rx equalization is optimized separately for each.

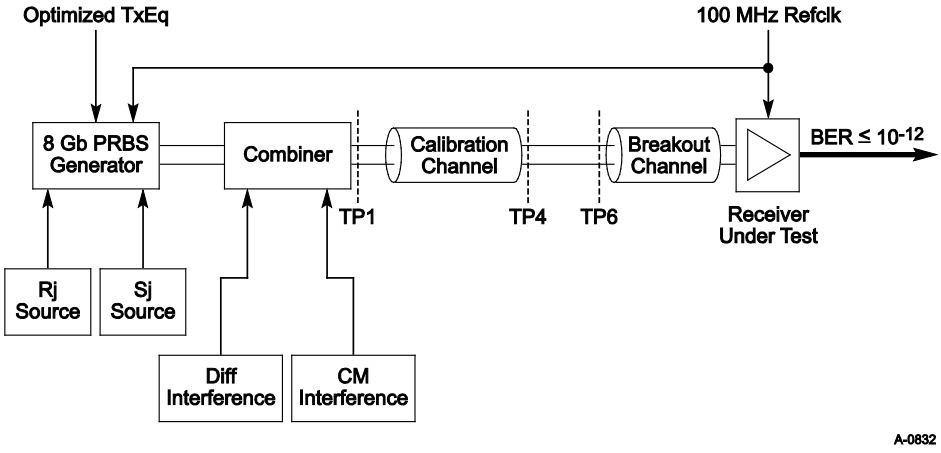


Figure 4-72: Layout for Stressed Voltage Testing of Receiver

4.3.4.4.3. Procedure for Calibrating a Stressed Jitter Eye

The stressed jitter calibration procedure is similar to that of stressed voltage. Only the long calibration channel (-20 dB) is used. Figure 4-73 illustrates the setup. Note that the same post processing scripts are applied identically as they are for the stressed voltage eye case. Eye width is fine tuned by making adjustments to the Rj source, while EH may be adjusted by varying the launch voltage at the generator.

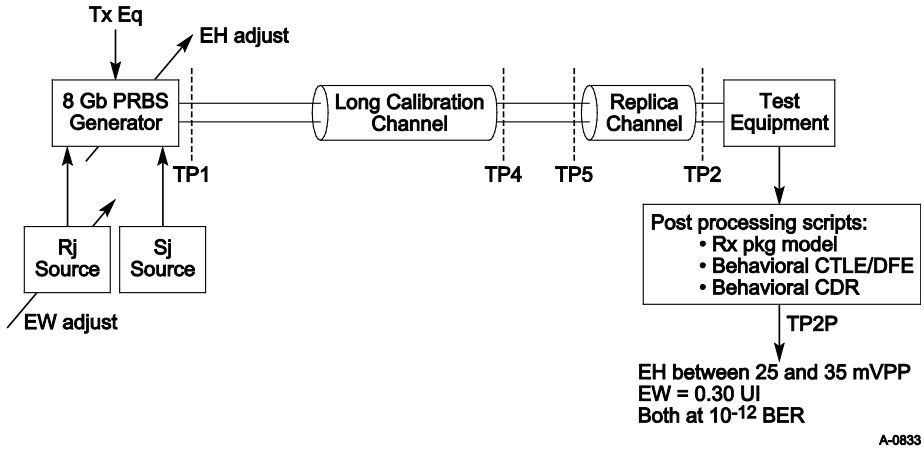


Figure 4-73: Layout for Calibrating the Stressed Jitter Eye

Table 4-23: Stressed Jitter Eye Parameters

Symbol	Parameter	Limits at 8.0 GT/s	Units	Comments
$V_{RX-LAUNCH-8G}$	Generator launch voltage	800 (nominal)	mVPP	Measured at TP1, see Figure 4-65. See Note 1.
$T_{RX-UI-8G}$	Unit Interval	125.00	ps	Nominal value is sufficient for Rx tolerancing. Value does not account for SSC.
$V_{RX-ST-8G}$	Eye height at TP2P	25 (min) 35 (max)	mVPP	At BER=10^{-12}. See Note 2.
$T_{RX-ST-8G}$	Eye width at TP2P	0.30	UI	At BER=10^{-12}. See Note 2.
$T_{RX-ST-SJ-8G}$	Sinusoidal Jitter	0.1 – 1.0	UI PP	See Figure 4-74 Measured at TP1. See Note 3.
$T_{RX-ST--RJ-8G}$	Random Jitter	3.0	ps RMS	Rj spectrally flat before filtering. Measured at TP1. See Note 4.

Notes:

1. $V_{RX-LAUNCH-8G}$ may be adjusted to meet $V_{RX-ST-8G}$ as long as the outside eye voltage at TP2 does not exceed 1300 mVPP.
2. $V_{RX-ST-8G}$ and $T_{RX-ST-8G}$ are referenced to TP2P and are obtained after post processing data captured at TP2. $V_{RX-ST-8G}$ and $T_{RX-ST-8G}$ include the effects of applying the behavioral Rx model and Rx behavioral equalization.
3. $T_{RX-ST-SJ-8G}$ may be measured at either TP1 or TP2.
4. While the nominal value is specified at 3.0 ps RMS, it may be adjusted to meet the 0.3 UI value for $T_{RX-ST-8G}$. Rj is measured at TP1 to prevent data-channel interaction from adversely affecting the accuracy of the Rj calibration.

 Rj is applied over the following range. The low frequency limit may be between 1.5 and 10 MHz, and the upper limit is 1.0 GHz.

4.3.4.4.4. Testing Receivers for Stressed/Swept Jitter

Once a calibrated EH and EW have been obtained, the cables are moved to connect the Rx DUT to the far end of calibration channel. The Tx equalization is then optimized as it was for the stressed voltage eye with the assumption that the DUT Rx will also optimize its equalization. Sj is set to an initial value that permits the receiver CDR to achieve lock. Then the Sj is swept over the frequency range shown in Figure 4-74, while keeping the Tx equalization fixed. A receiver must meet or be below the 10^{-12} BER over the entire swept Sj range.

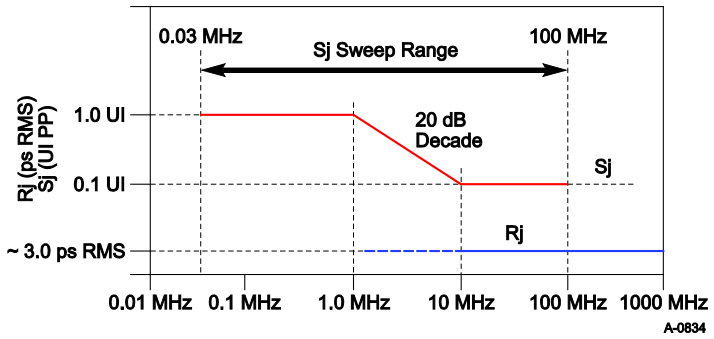


Figure 4-74: Swept Sj Mask

4.3.4.4.5. Data Clocked and Common Refclk Rx Architectures

The testing of data clocked and common Refclk Rx architectures is nearly identical, except for the characteristics of the 100 MHz Refclk. The stressed eye calibration step is identical for the two cases, except for the common Refclk (see Figure 4-75) case the generator and DUT share a common Refclk. For the data clocked case, the DUT and the generator are driven with an SSC modulated Refclk (see Figure 4-76). SSC shall have a triangular modulation profile and meet the frequency limits and maximum deviation range defined in Table 4-37.

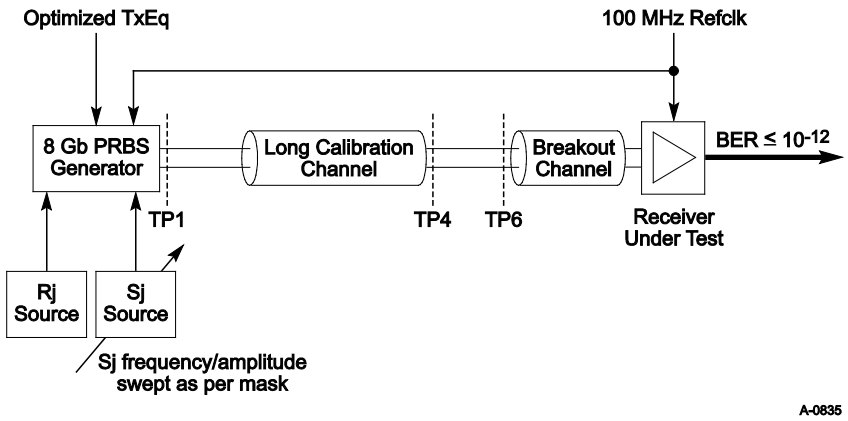


Figure 4-75: Layout for Jitter Testing Common Refclk Rx

The dotted line between the Rx DUT and the SSC modulated Refclk in Figure 4-76 indicates that the Refclk is not generally required to derive the receiver's data clock, but it may be required for other purposes and must therefore be supplied to the DUT. Note that the BERT's modulation bandwidth must be sufficient to pass the SSC through to its output. This translates to a modulation BW of at least 1.5 MHz.

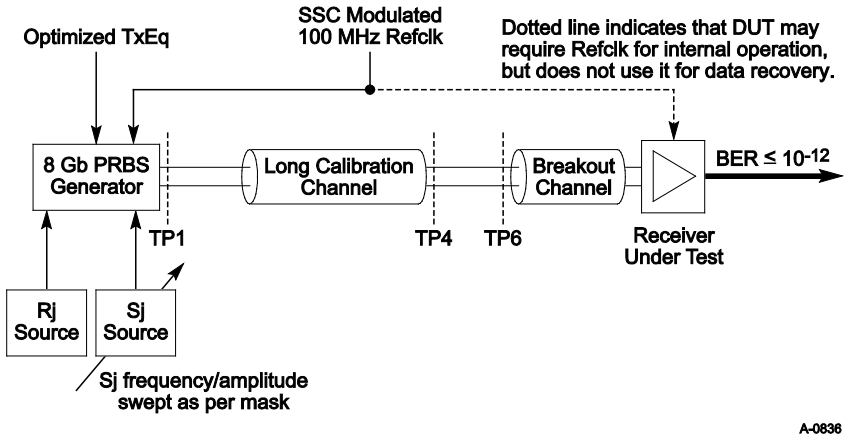


Figure 4-76: Layout for Jitter Testing Data Clocked Refclk Rx

4.3.4.4.6. Separate Refclk with Independent SSC (SRIS) RX Architecture

Comment [VS152]: ECN: SRIS

The testing of the Separate Refclk with Independent SSC RX Architecture is the same as the Common clock architecture with the following modifications:

- The DUT shall be driven with an independent SSC modulated reference clock for all testing. SSC shall meet the frequency limits and maximum deviation range defined in Table 4-37.
- The generator shall be driven with an independent SSC modulated reference clock for the stressed voltage test. SSC shall have a triangular profile and meet the frequency limits and maximum deviation range defined in Table 4-37.
- The generator shall provide a 33 kHz Sj tone with 25 ns peak to peak amplitude during the stressed eye test.
- The generator shall transmit with a nominal unit interval of 125 ps + 2500 PPM during the stressed eye test.
- Sj shall be swept over the range shown in Figure 4-77 during the stressed eye test:

$$\textbf{Sj in Unit Intervals} = \begin{cases} \frac{5713941}{F^{1.1262}} & (\textbf{400 KHz} \leq F < \textbf{1 MHz}) \\ \frac{10^6}{F} & (\textbf{1 MHz} \leq F < \textbf{10 MHz}) \\ \textbf{0.1} & (F \geq \textbf{10 MHz}) \end{cases}$$

Equation 4.3.3

Comment [VS153]: D-ECN___SRIS_JTOL_SSC__31_March_14

where F is frequency in Hz

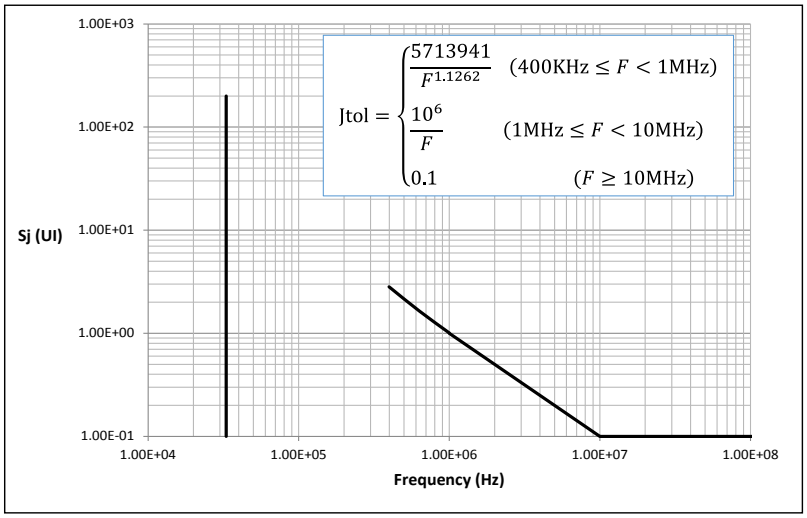


Figure 4-77: Swept Sj Mask

4.3.4.5. Common Receiver Parameters

The following table defines the parameters for 2.5 GT/s, 5.0 GT/s, and 8.0 GT/s Receivers.

Table 4-24: Receiver Specifications

Symbol	Parameter	2.5 GT/s	5.0 GT/s	8.0 GT/s	Units	Comments
UI	Unit Interval	399.88 (min) 400.12 (max)	199.94 (min) 200.06 (max)	124.9625 (min) 125.0375 (max)	ps	UI does not account for SSC caused variations.
$V_{RX-DIFF-PP-CC}$	Differential Rx peak-peak voltage for common Refclk Rx architecture	0.175 (min) 1.2 (max)	0.120 (min) 1.2 (max)	See Table 4-22 and Table 4-23.	V	See Section 4.3.7.3.2.
$V_{RX-DIFF-PP-DC}$	Differential Rx peak-peak voltage for data clocked Rx architecture	0.175 (min) 1.2 (max)	0.100 (min) 1.2 (max)	See Table 4-22 and Table 4-23.	V	See Section 4.3.7.3.2.
T_{RX-EYE}	Receiver eye time opening	0.40 (min)	See Table 4-20 and Table 4-21.	See Table 4-22 and Table 4-23.	UI	Minimum eye time at Rx pins to yield a 10^{-12} BER. See Note 1.
$T_{RX-TJ-CC}$	Max Rx inherent timing error	N/A	0.40 (max)	See Table 4-22 and Table 4-23.	UI	Max Rx inherent total timing error for common Refclk Rx architecture. See Note 2.
$T_{RX-TJ-DC}$	Max Rx inherent timing error	N/A	0.34 (max)	See Table 4-22 and Table 4-23.	UI	Max Rx inherent total timing error for data clocked Rx architecture. See Note 2.
$T_{RX-DJ-DD-CC}$	Max Rx inherent deterministic timing error	N/A	0.30 (max)	See Table 4-22 and Table 4-23.	UI	Max Rx inherent deterministic timing error for common Refclk Rx architecture. See Note 2.
$T_{RX-DJ-DD-DC}$	Max Rx inherent deterministic timing error	N/A	0.24 (max)	See Table 4-22 and Table 4-23.	UI	Max Rx inherent deterministic timing error for data clocked Rx architecture. See Note 2.
$T_{RX-EYE-MEDIAN-to-MAX-JITTER}$	Max time delta between median and deviation from median	0.3 (max)	Not specified	Not specified	UI	Only specified for 2.5 GT/s.
$T_{RX-MIN-PULSE}$	Minimum width pulse at Rx	Not specified	0.6 (min)	Not specified	UI	Measured to account for worst Tj at 10^{-12} BER. See Figure 4-38.
$V_{RX-MAX-MIN-RATIO}$	min/max pulse voltage on consecutive UI	Not specified	5 (max)	Not specified	--	Rx eye must simultaneously meet V_{RX-EYE} limits.
BW_{RX-PLL}	Rx PLL BW for 2.5 GT/s	22 (max) 1.5 (min)	Not specified	Not specified	MHz	See Note 3.

Symbol	Parameter	2.5 GT/s	5.0 GT/s	8.0 GT/s	Units	Comments
$BW_{RX-PKG-PLL1}$	Rx PLL bandwidth corresponding to $PKG_{RX-PLL1}$	Not specified	16 (max) 8 (min)	4 (max) 2 (min)	MHz	Second order PLL jitter transfer bounding function. See Note 3.
$BW_{RX-PKG-PLL2}$	Rx PLL bandwidth corresponding to $PKG_{RX-PLL2}$	Not specified	16 (max) 5 (min)	5 (max) 2 (min)	MHz	Second order PLL jitter transfer bounding function. See Note 3.
$PKG_{RX-PLL1}$	Rx PLL peaking limit #1	Not specified	3.0 (max)	2.0 max	dB	PLL BW = 8 MHz (min) @ 5.0 GT/s or BW = 4 MHz (max) @ 8.0 GT/s. See Note 3.
$PKG_{RX-PLL2}$	Rx PLL peaking limit #2	Not specified	1.0 (max)	1.0 (max)	dB	PLL BW = 5 MHz (min) @ 5.0 GT/s or BW = 5 MHz (max) @ 8.0 GT/s. See Note 3.
$RL_{RX-DIFF}$	Rx package plus Si differential return loss	10 (min)	10 (min) for 0.05 - 1.25 GHz 8 (min) for >1.25 - 2.5 GHz	10 (min) for 0.05 - 1.25 GHz 8 (min) for >1.25 - 2.5 GHz 5 (min) for >2.5 - 4. GHz	dB	See Figure 4-60 and Note 4.
RL_{RX-CM}	Common mode Rx return loss	6 (min)	6 (min)	6 (min) for 0.05 - 2.5 GHz 5 (min) for >2.5 - 4.0 GHz	dB	See Figure 4-60 and Note 4.
Z_{RX-DC}	Receiver DC single ended impedance	40 (min) 60 (max)	40 (min) 60 (max)	Not specified	Ω	DC impedance limits are needed to guarantee Receiver detect. For 8.0 GT/s is bounded by RL_{RX-CM}. See Note 5.
T_{RX-GND_FLOAT}	Rx termination ground float time	Not specified	Not specified	500	µs	Time allowed to float Rx internal ground in 2.5 GT/s/5.0 GT/s to 8.0 GT/s configuration change. See Note 8.
$Z_{RX-DIFF-DC}$	DC differential impedance	80 (min) 120 (max)	Not specified	Not specified	Ω	For 5.0 GT/s and 8.0 GT/s covered under the $RL_{RX-DIFF}$ parameter. See Note 5.
$V_{RX-CM-AC-P}$	Rx AC common mode voltage	150 (max)	150 (max)	75 mV (max) (EH <100 mVPP) 125 mV (max) (EH ≥100 mVPP) See Table 4-22.	mVP	Measured at Rx pins into a pair of 50 Ω terminations into ground. See Note 6.

Symbol	Parameter	2.5 GT/s	5.0 GT/s	8.0 GT/s	Units	Comments
$Z_{RX-HIGH-IMP-DC-POS}$	DC Input CM Input Impedance for V>0 during Reset or power down	≥10 k or ≥20 k	≥10 k or ≥20 k	≥10 k or ≥20 k	Ω	Rx DC CM impedance with the Rx terminations not powered. ≥10 kΩ over 0-200 mV range wrt. ground and ≥20 kΩ for voltages ≥200 mV wrt ground. See Note 9.
$Z_{RX-HIGH-IMP-DC-NEG}$	DC Input CM Input Impedance for V<0 during Reset or power down	1.0 k (min)	1.0 k (min)	1.0 k (min)	Ω	Rx DC CM impedance with the Rx terminations not powered, measured over the range -150 - 0 mV with respect to ground. See Note 7.
$V_{RX-IDLE-DET-DIFFp-p}$	Electrical Idle Detect Threshold	65 (min) 175 (max)	65 (min) 175 (max)	65 (min) 175 (max)	mV	$V_{RX-IDLE-DET-DIFFp-p} = 2*\|V_{RX-D+} - V_{RX-D-}\|$. Measured at the package pins of the Receiver. See Section 4.2.4.3.
$T_{RX-IDLE-DET-DIFF-ENTERTIME}$	Unexpected Electrical Idle Enter Detect Threshold Integration Time	10 (max)	10 (max)	10 (max)	ms	An unexpected Electrical Idle ($V_{RX-DIFF-PP} < V_{RX-IDLE-DET-DIFFp-p}$) must be recognized no longer than $T_{RX-IDLE-DET-DIFF-ENTERTIME}$ to signal an unexpected idle condition.
$L_{RX-SKEW}$	Lane to Lane skew	20 (max)	8 (max)	6 (max)	ns	Across all Lanes on a Port. $L_{RX-SKEW}$ comprehends Lane-Lane variations due to channel and repeater delay differences.

Notes:

1. Receiver eye margins are defined into a 2 x 50 Ω reference load.
2. The four inherent timing error parameters are defined for the convenience of Rx designers, and they are measured during Receiver tolerancing.
3. Two combinations of PLL BW and peaking are specified at 5.0 GT/s to permit designers to make tradeoffs between the two parameters. If the PLL's min BW is ≥8 MHz, then up to 3.0 dB of peaking is permitted. If the PLL's min BW is relaxed to ≥5.0 MHz, then a tighter peaking value of 1.0 dB must be met. Note: a PLL BW extends from zero up to the value(s) defined as the min or max in the above table. For 2.5 GT/s a single PLL bandwidth and peaking value of 1.5-22 MHz and 3.0 dB are defined.
4. Measurements must be made for both common mode and differential return loss. In both cases the DUT must be powered up and DC isolated, and its D+/D- inputs must be in the low-Z state.
5. The Rx DC single ended impedance must be present when the Receiver terminations are first enabled to ensure that the Receiver Detect occurs properly. Compensation of this impedance can start immediately and the Rx Common Mode Impedance (constrained by RL_{RX-CM} to 50 Ω ±20%) must be within the specified range by the time Detect is entered.
6. Common mode peak voltage is defined by the expression: max{|(Vd+ - Vd-) - V_{-CMDC}|}.
7. $Z_{RX-HIGH-IMP-DC-NEG}$ and $Z_{RX-HIGH-IMP-DC-POS}$ are defined respectively for negative and positive voltages at the input of the Receiver. Transmitter designers need to comprehend the large difference between >0 and <0 Rx impedances when designing Receiver detect circuits.
8. Defines the time for the receiver's input pads to settle to new common-mode on 2.5 GT/s/5.0 GT/s transition to 8.0 GT/s.
9. For voltages >500 mV the effects of Rx ESD structures may limit $Z_{RX-HIGH-IMP-DC-POS}$ to values less than 20 KΩ.

4.3.4.5.1. 5.0 GT/s Exit From Idle Detect (EFI)

It is difficult to scale the capabilities of the EFI detect circuits with data rate, and for that reason the 5.0 GT/s specification defines different data patterns in the FTS and the TS1 and TS2 Ordered Sets than are defined for 2.5 GT/s operation. In particular, repeated K28.7 patterns are defined to guarantee sufficient voltage and time requirements, as illustrated in the figure below. Concatenated EIE Symbols yield alternating one/zero run lengths of five UI each.

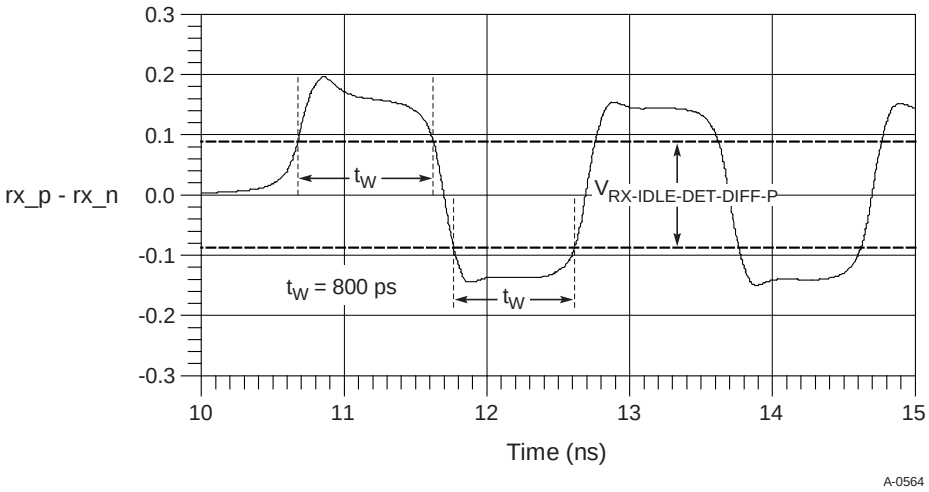


Figure 4-78: Exit from Idle Voltage and Time Margins

4.3.4.5.2. Receiver Loopback

Receivers capable of generating the Modified Compliance Pattern must implement Error Count described in Section 4.2.9.

4.3.4.5.3. Receiver Return Loss

The measurement methodology and frequency binning for differential and common mode Rx RL is identical to that for the Tx. The only difference exists between the 2.5-4.0 GHz differential RL limits. For details refer to Figure 4-56 and Figure 4-57.

4.3.5. Low Frequency and Miscellaneous Signaling Requirements

The parameters defined in this section are relevant to 2.5, 5.0, and 8.0 GT/s designs.

4.3.5.1. Channel AC Coupling Capacitors

Each Lane of a PCI Express Link must be AC coupled. The minimum and maximum values for the capacitance is given in Table 4-18. Capacitors must be placed on the Transmitter side of an interface that permits adapters to be plugged and unplugged. In a topology where everything is located on a single substrate, the capacitors may be located anywhere along the channel. External capacitors are assumed because the values required are too large to feasibly construct on-chip.

4.3.5.2. Short Circuit Requirements

All Transmitters and Receivers must support surprise hot insertion/removal without damage to the component. The Transmitter and Receiver must be capable of withstanding sustained short circuit to ground of D+ and D-.

4.3.5.3. Transmitter and Receiver Termination

- The Transmitter is required to meet $RL_{TX-DIFF}$ and RL_{TX-CM} (see Table 4-18, Figure 4-56, and Figure 4-57) any time functional differential signals are being transmitted.
- The Transmitter is required only to meet $I_{TX-SHORT}$ (see Table 4-18) any time functional differential signals are not being transmitted.
- Note: The differential impedance during this same time is not defined.
- The Receiver is required to meet $RL_{RX-DIFF}$ and RL_{RX-CM} (see Table 4-24) during all LTSSM states excluding only times during when the device is powered down, Fundamental Reset is asserted, or when explicitly specified.
- The Receiver is required to meet $Z_{RX-HIGH-IMP-DC-NEG}$ and $Z_{RX-HIGH-IMP-DC-POS}$ (see Table 4-24) any time adequate power is not provided to the Receiver, Fundamental Reset is asserted, or when explicitly specified.

4.3.5.4. Electrical Idle

Electrical Idle is a steady state condition where the Transmitter D+ and D- voltages are held constant at the same value. Electrical Idle is primarily used in power saving and inactive states (e.g., Disabled).

Before a Transmitter enters Electrical Idle, it must always send the required number of EIOSs defined in Section 4.2.4.2 except for the LTSSM substates explicitly exempted from this requirement. After sending the last Symbol of the last of the required number of EIOSs, the Transmitter must be in a valid Electrical Idle state within the time as specified by $T_{TX-IDLE-SET-TO-IDLE}$ in Table 4-18.

The successful reception of an EIOS occurs based on the rules in Section 4.2.4.2. It must be noted that in substates (e.g., Loopback.Active for a loopback slave) where multiple consecutive EIOSs are expected, the Receiver must receive the appropriate number of EIOS sequences comprising of COM, IDL, IDL, IDL.

The low impedance common mode and differential Receiver terminations values (see Table 4-18 and Table 4-24) must be met in Electrical Idle. The Transmitter can be in either a low or high impedance mode during Electrical Idle.

Any time a Transmitter enters Electrical Idle it must remain in Electrical Idle for a minimum of $T_{TX\text{-}IDLE\text{-}MIN}$. The Receiver should expect the last EIOS followed by a minimum amount of time in Electrical Idle ($T_{TX\text{-}IDLE\text{-}MIN}$) to arm its Electrical Idle Exit detector.

When the Transmitter transitions from Electrical Idle to a valid differential signal level it must meet the output return loss specifications described in Figure 4-56 and Figure 4-57.

Electrical Idle Exit shall not occur if a signal smaller than $V_{RX\text{-}IDLE\text{-}DET\text{-}DIFFp\text{-}p}$ minimum is detected at a Receiver. Electrical Idle Exit shall occur if a signal larger than $V_{RX\text{-}IDLE\text{-}DET\text{-}DIFFp\text{-}p}$ maximum is detected at a Receiver. Electrical Idle may be detected on the received signal regardless of its frequency components, or it may be detected only when the received signal is switching at a frequency of 125 MHz or higher.

4.3.5.5. DC Common Mode Voltage

The Receiver DC common mode voltage is nominally 0 V when operating at 2.5 GT/s or 5.0 GT/s.

The Transmitter DC common mode voltage is held at the same value during all states unless otherwise specified. The range of allowed Transmitter DC common mode values is specified in Table 4-18 ($V_{TX\text{-}DC\text{-}CM}$).

4.3.5.6. Receiver Detection

The Receiver Detection circuit is implemented as part of a Transmitter and must correctly detect whether a load impedance equivalent to a DC impedance implied by the $Z_{RX\text{-}DC}$ parameter (40 Ω-60 Ω) is present. Note: Support for Rx detect, which only occurs at 2.5 GT/s, is the reason why 2.5 GT/s Receivers impedance at DC is specified.

The recommended behavior of the Receiver Detection sequence is described below:

Step 1. A Transmitter must start at a stable voltage prior to the detect common mode shift.

Step 2. A Transmitter changes the common mode voltage on D+ and D- consistent with meeting the $V_{TX\text{-}RCV\text{-}DETECT}$ parameter and consistent with detection of Receiver high impedance which is bounded by parameters $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}POS}$, $Z_{RX\text{-}HIGH\text{-}IMP\text{-}DC\text{-}NEG}$, and Note 7 in Table 4-24.

Step 3. A Receiver is detected based on the rate that the lines change to the new voltage.

a. The Receiver is not present if the voltage at the Transmitter charges at a rate dictated only by the Transmitter powered-off impedance and the capacitance of the interconnect and series capacitor.

b. The Receiver is present if the voltage at the Transmitter charges at a rate dictated by the Transmitter powered-off impedance, the series capacitor, the interconnect capacitance, and the Receiver termination.

Upon coming out of Rx detect a device must meet all the specification parameters required for 2.5 GT/s operation.

Any time Electrical Idle is exited the detect sequence does not have to execute or may be aborted on that Lane.

4.3.5.6.1. Differential Receiver Detect

If an implementation chooses to transition from Detect to Polling based on Electrical Idle being broken prior to performing a Receiver detect sequence, an unreliable Link could be formed; due to the possibility that there may not be a low impedance termination resistor on both Rx differential conductors, which make up the differential pair.

If the Receiver detection circuit performs the detect sequence on each conductor of the differential pair (both D+ and D-) and detects a load impedance greater than $Z_{RX\text{-}DC}$ on either conductor, the Receiver detection circuit shall interpret this as no termination load present and respond as if neither load were present.

In this revision of this specification, it is not required that the detect sequence be performed on both conductors of a differential pair. Future revisions of this specification may require the successful detection of a low impedance termination resistor on both differential pairs prior to transitioning to Polling.

4.3.5.7. Beacon

Support for Beacon is required for all PCI Express components that support a wakeup mechanism in order to function in form factors that require the use of Beacon. However, not all systems and form factor specifications require the use of Beacon, and for components that are restricted to use in such environments it is not necessary to support Beacon.

Note: Devices operating at 5.0 GT/s or 8.0 GT/s do not need to support Beacon.

This section applies to all components that support Beacon.

The Beacon is a signal sent by a Downstream component to start the exit from an L2 state.

All Transmitter electrical specifications (Table 4-18) must be met while sending a Beacon with the following exceptions and clarifications.

- The Beacon is a DC Balanced signal of periodic arbitrary data, which is required to contain some pulse widths >= 2 ns but no larger than 16 µs.
- The maximum time between qualifying pulses (2 ns <= x <= 16 µs) can be no longer than 16 µs.
- DC Balance must be always be restored within a maximum time of 32 µs.
- Beacon is transmitted in a low impedance mode.
- All Beacons must be transmitted and received on at least Lane 0 of multi-Lane Links.[60]
- The output Beacon voltage level must be at a -6 dB de-emphasis level for Beacon pulses with a width greater than 500 ns.
- The output Beacon voltage level can range between a specified voltage level (see $V_{TX-DIFF-PP}$ in Table 4-18) and a corresponding -3.5 dB de-emphasized voltage levels for Beacon pulses smaller than 500 ns.
- The Lane-to-Lane Output Skew (see $L_{TX-SKEW}$ in Table 4-18) and SKP Ordered Set output (see Section 4.2.7) specifications do not apply.
- When any Bridge and/or Switch receives a Beacon at a Downstream Port, that component must propagate a Beacon wakeup indication Upstream. This wakeup indication must use the appropriate wakeup mechanism required by the system or form factor associated with the Upstream Port of the Switch (see Section 5.3.3.2).

For the case described above of Beacon pulses with a width greater than 500 ns, the minimum Beacon amplitude is -6 dB down from the minimum differential peak to peak output voltage ($V_{TX-DIFF-PP}$). The maximum Beacon amplitude for this case is -6 dB down from the maximum peak to peak output voltage ($V_{TX-DIFF-PP}$).

4.3.5.7.1. Beacon Example

An example Receiver waveform driven at the -6 dB level for a 30 kHz Beacon is shown in Figure 4-79. An example Receiver waveform using the COM Symbol at full speed signaling is shown in Figure 4-80. It should be noted that other waveforms and signaling are possible other than the examples shown in Figure 4-79 and Figure 4-80 (i.e., Polling is another valid Beacon signal).

[60] Lane 0 as defined after Link width and Lane reversal negotiations are complete.

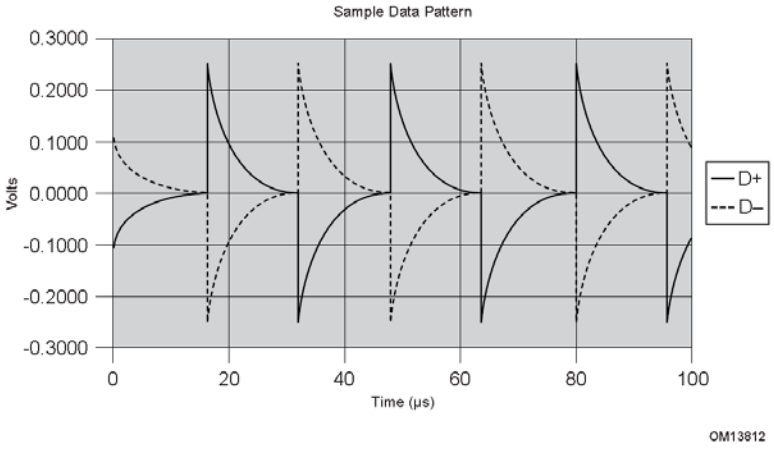


Figure 4-79: A 30 kHz Beacon Signaling Through a 75 nF Capacitor

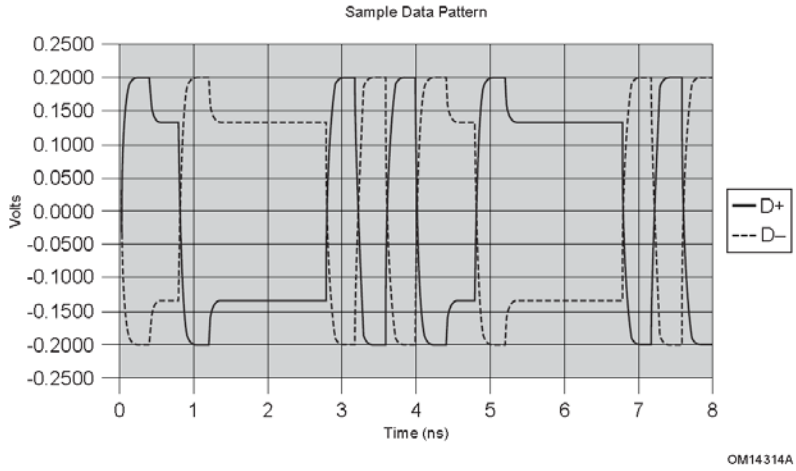


Figure 4-80: Beacon, Which Includes a 2-ns Pulse Through a 75 nF Capacitor

4.3.6. Channel Specification

4.3.6.1. Channel Validation at 2.5 GT/s and 5.0 GT/s

Channel loss (attenuation of the differential voltage swing) in this system is a critical parameter that must be properly considered and managed in order to ensure proper system functionality. Channel loss is specified in terms of the amount of attenuation or loss that can be tolerated between the Transmitter and Receiver. The Tx is responsible for producing the specified differential eye height at the pins of its package. Together, the Tx, the Refclk, and the interconnect are responsible for producing the specified differential eye height and width at the Rx pins (see Figure 4-63).

An approximate way to understand the worst-case operational loss budget at 1.25 GHz is calculated by taking the minimum output voltage ($V_{TX-DIFF-PP}$ = 800 mV) divided by the minimum input voltage to the Receiver ($V_{RX-DIFF-PP}$ = 175 mV), which results in a maximum loss of 13.2 dB. The approximate way to understand the worst-case operational loss budget at 625 MHz is calculated by taking the minimum de-emphasized output voltage ($V_{TX-DIFF-PP}$ = 505 mV) divided by the minimum input voltage to the Receiver ($V_{RX-DIFF-PP}$ = 175 mV), which results in a maximum loss of 9.2 dB. Although loss vs. frequency is useful in understanding how to design an effective interconnect, the timing and voltage margin measured in the Tx and Rx eye diagrams end up being the ultimate constraints of insertion loss.

4.3.6.2. Channel Characteristics at 5.0 GT/s

At 5.0 GT/s a more accurate method of comprehending the effects of channel loss is required in order to avoid excessive guardbanding. The method described here imports the channel's s-parameters into a simulation environment that includes worst case models for Transmitters and data patterns. The resulting time domain simulation yields eye diagrams from which voltage and timing margins may be obtained and compared against those defined for the Receiver. Note: The methodology described in Sections 4.3.6.2 through 4.3.6.2.7 must be applied to 5.0 GT/s designs, and may be applied to 2.5 GT/s designs.

A channel's characteristics are completely defined by its s-parameters, in particular: insertion loss, return loss, and aggressor-victim coupling. It can been demonstrated that these parameters are sufficient to completely quantify all channel-induced phenomena affecting eye margins including: I/O-channel impedance mismatch, insertion loss, jitter amplification, impedance discontinuities, and crosstalk. Long channels tend to be dominated by insertion loss and crosstalk, while short channels tend to dominated by impedance discontinuities. Since both types of channels are possible in PCI Express implementations, it is necessary to provide a means of characterizing the channel that comprehends all possible channel characteristics.

It is not, in general, possible to establish a straightforward correlation, easily expressed in graphical terms, between channel s-parameters and eye margins for all types of channels. For example plotting $|s_{21}|$ vs. frequency does not give useful correlations to eye margins for all possible channels. Instead it is necessary to convolve the channel's s-parameters with a worst case Transmitter behavioral model and a worst case data pattern. The resulting time domain results, represented via an eye diagram, can then be compared against the V_{RX-EYE} and T_{RX-EYE} parameters defined in the Receiver specification. The following sections detail this procedure.

4.3.6.2.1. Procedure for Channel Measurement and Margin Extraction

A channel may be characterized to specification by means of the following procedure.

1. s-parameters for the channel under test are measured. These include insertion loss, return loss, and (if applicable) crosstalk.
2. The s-parameters are imported into a simulation environment along with a reference load and a behavioral Transmitter model.
3. Worst case Transmitter corners and data patterns are driven through the channel model into a reference load. Then the resulting eye margins are post processed to account for Refclk and Transmitter jitter effects that were not included in the simulation.

4. The resulting eye margins are compared to T_{RX-EYE} in Table 4-24 and V_{RX-EYE} in Table 4-20 and Table 4-21 to determine if the channel is within specification.

The following sections detail the four steps listed above.

4.3.6.2.2. Acquiring the Channel's s-parameters

Channel s-parameter data should be taken over the entirety of the 5.0 GT/s PCI Express signaling spectral range which extends from approximately 250 MHz to 20 GHz or greater. Both near and far end return loss as well as forward and reverse insertion loss should be measured, since most simulation tools require a complete s-parameter representation.

4.3.6.2.3. Defining the Simulation Environment

An end-to-end simulation environment, as shown in Figure 4-81, is necessary to generate eye margins. The model consists of the following components: behavioral Tx model, channel s-parameters, reference load, and data pattern(s). For a topology with minimal crosstalk, a single-Lane model will suffice. Otherwise the model needs to include aggressor and victim components.

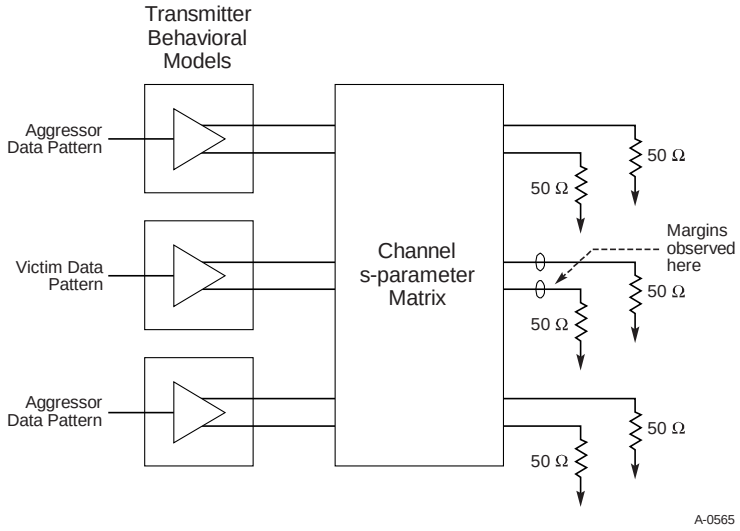


Figure 4-81: Simulation Environment for Characterizing Channel

4.3.6.2.4. Defining Worst Case Data Patterns and Tx Corners

The behavioral Tx model must support adjustable parameters corresponding to those listed in Table 4-25. Stressing the channel to yield worst case margins at the far end requires that a combination of worst case Tx parameters and data patterns be simulated. In most cases a parameter may be set to its worst case corner by inspection. For example, $V_{TX-DIFF-PP}$ will yield the worst case eye margins when set to a minimum. However, some parameters will need to be simulated over

their min to max range in order to guarantee worst case voltage/time margins for a particular channel.

Figure 4-81 shows all transmitting Lanes routed in a non-interleaved fashion, and this topology is recommended, because interleaved routing yields unacceptably high levels of near-end crosstalk for the routing guidelines typically encountered in PC-type platforms. By non-interleaved it is meant that all Tx Lanes are routed as a group. Interleaved indicates that Tx and Rx Lanes are routed in an alternating fashion.

In addition to the above mentioned Rx parameters, channel simulation must generate data patterns yielding worst case margins while conforming to 8b/10b encoding rules. Two approaches may be used: a fixed pattern or a channel specific pattern. The former approach is simpler, but may not yield as much pattern dependent margin degradation as the latter.

Operation at 5.0 GT/s supports Selectable De-emphasis, where the value of -3.5 or -6 dB is selected upon power-up. For a given channel, it is assumed that the developer would know a priori what de-emphasis value is optimum, and that value would be included in the simulation.

Table 4-25: Worst Case Tx Corners for 5.0 GT/s Channel Simulation

Symbol	Parameter	Min	Max	Units	Notes
$V_{TX-DIFF-PP}$	Differential p-p Tx voltage swing	800		mV	Only minimum value needed in simulations.
$V_{TX-DIFF-PP-LOW}$	Low power differential p-p Tx voltage swing	400		mV	Only minimum value needed in simulations. See Section 4.3.6.2.6.
$V_{TX-DE-RATIO-6DB}$	Tx de-emphasis level	5.5	6.5	dB	Both minimum/maximum range needs to be simulated. See Note 1.
$V_{TX-DE-RATIO-3.5DB}$	Tx de-emphasis level	3.0	4.0	dB	Both minimum/maximum range needs to be simulated. See Note 1.
$T_{MIN-PULSE}$	Instantaneous pulse width	0.9		UI	Single pulses of both polarity need to be compressed to minimum value.
$T_{TX-RISE-FALL}$	Transmitter rise and fall time	0.15		UI	Some channels are sensitive to minimum value. Maximum value defined by T_{TX-EYE}.
$T_{RF-MISMATCH}$	Tx rise/fall mismatch		0.1	UI	Only maximum value needed in simulations.
$RL_{TX-DIFF}$	Tx package plus Si differential return loss		See Figure 4-78.		See Note 2.
RL_{TX-CM}	Tx package plus Si common mode return loss		6	dB	Same as defined in Tx specification.

Symbol	Parameter	Min	Max	Units	Notes
$T_{CH-TX-DJ-SIM}$	Maximum jitter for channel driven by Tx with 0.1 UI DCD		78.1	ps	Assumes Tx is simulated with max of 0.1 UI DCD with no other Tx jitter components. See Figure 4-82A.
$T_{EYE-SLEW-CC}$	Amount by which eye is slewed to obtain voltage margin for common Refclk Rx architecture		±28.9	ps	Resulting eye voltage margin must meet $V_{RX-DIFF-PP-CC}$ in Table 4-24.
$T_{EYE-SLEW-DC}$	Amount by which eye is slewed to obtain voltage margin for data clocked Rx architecture		±34.7	ps	Resulting eye voltage margin must meet $V_{RX-DIFF-PP-DC}$ in Table 4-24.

Notes:

1. Two Tx de-emphasis ratios are defined to account for selectable Tx de-emphasis based on channel characteristics. Usually the de-emphasis ratio is preset based on a priori knowledge of which value is optimum for the channel the Tx is driving, and that de-emphasis ratio is used to simulate the channel's characteristics.
2. Typically a package is modeled as a combination of the Tx Si and the package interconnect, where the Si may be defined as a voltage or current source with a particular impedance driving a parasitic capacitance. The package interconnect may be modeled as t-line elements. The combination of the above elements is then adjusted to yield one or more worst case impedance profiles that fit under the Tx return loss curves. Note that more than one package/Si model may be needed since it is not generally possible to synthesize a package/Si model that replicates the entire Tx return loss profile.

4.3.6.2.5. Accounting for Jitter When Characterizing the Channel

It is most convenient to simulate the channel without introducing low frequency jitter components in either the Refclk or the Transmitter. These jitter components have been shown to propagate through the channel on a ps for ps basis. Pulse width compression, acting on single pulses sourced by the Transmitter, does need to be included in the simulation, since it interacts with the channel and produces so called jitter amplification. While they need not be included in the actual simulation, low frequency jitter components must be accounted for in the overall jitter budgeting process. This will require that we deduct the worst case low frequency jitter components from the Refclk and the Transmitter from the simulated eye margins, before comparing against the Receiver eye parameters.

As mentioned above, the timing and voltage margins obtained when simulating the channel model need only include high frequency Tx jitter responsible for compressing a pulse to a minimum width. However, suitable post processing must be applied before we can compare the results against the Rx jitter parameters:

Include the impact of non-simulated jitter > 1.5 MHz, but excluding DCD.

Determine the center of the eye in order to measure voltage and time margins.

Figure 4-82A illustrates the effect of simulating a channel with only DCD phase jitter included. The second step consists of replicating the effect that the non-simulated jitter components would have

on the eye, and this may be achieved by slewing the eye in the x-direction by an amount equal to ½ the Dj and Rj components that were not simulated. Note that since common Refclk Rx and data clocked Rx architectures specify different amounts of jitter, so the amount by which the simulated eye is slewed will differ. Once the composite eye has been generated the next step is to determine the eye center and obtain the eye voltage and time opening margins. Determining the eye center may be achieved by using the filter function applied to the Receiver. Then the margins may be compared against the $T_{RX\text{-}TJ\text{-}CC}$, $T_{RX\text{-}TJ\text{-}DC}$ and $V_{RX\text{-}DIFF\text{-}PP\text{-}CC}$, $V_{RX\text{-}DIFF\text{-}PP\text{-}DC}$, as appropriate for the common Refclk Rx and the data clocked Rx architectures.

When simulating Tx jitter, we have assumed that the maximum amount of DCD (0.1 UI) is being applied. The difference between the maximum $T_{TX\text{-}HF\text{-}DJ\text{-}DD}$ (0.15 UI) and the above 0.1 UI represents the amount of additional Tx Djdd that is not simulated, or 10 ps. Similarly, Rj terms from the Tx and the Refclk must be comprehended by RSSing together the maximum Tx Rj (1.4 ps RMS) and the maximum Refclk Rj (3.1 ps RMS for the common Refclk Rx architecture and 4.0 ps RMS for the data clocked Rx architecture). The resulting terms ($T_{NON\text{-}SIM\text{-}JITTER\text{-}TJ}$) define the amount by which the simulated eye must be slewed: ± 28.9 ps for the common Refclk Rx architecture and ± 34.7 ps for the data clocked Rx architecture.

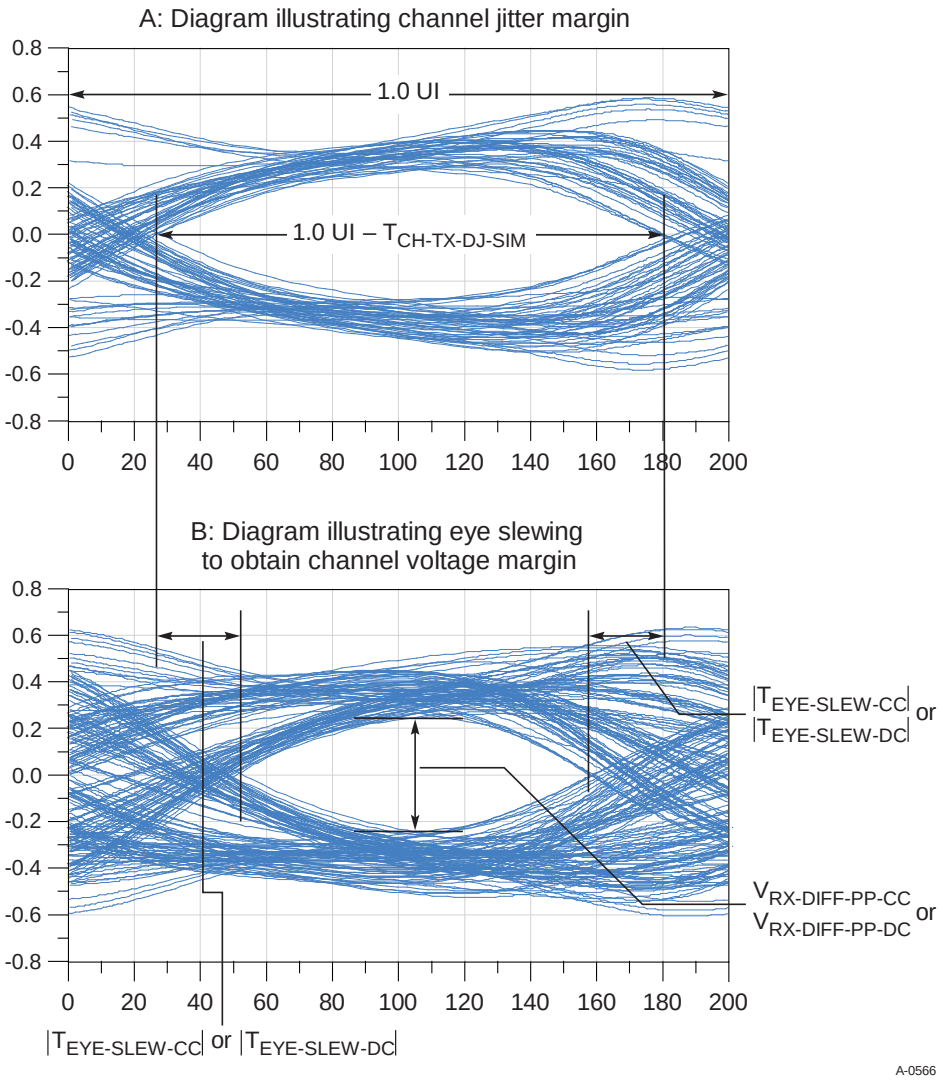


Figure 4-82: Extracting Eye Margins from Channel Simulation Results

4.3.6.2.6. Specifying the Channel for the Reduced Swing Option

Certain PCI Express applications that are power sensitive, such as mobile, may be implemented using reduced swing Transmitter option. This involves the use of a reduced swing transmit signal with no de-emphasis. The procedure for specifying a channel for the reduced swing Transmitter is identical to that used for the full swing Transmitter, with the exception that the worst case behavioral Tx characteristics must reflect the reduced swing and lack of de-emphasis.

4.3.6.2.7. Multi-Segment Channels

Multi-segment channels may be specified in a manner similar to that applied to single segment channels in the preceding sections. Different platform topologies will need specific partition budgets, depending on the ratio of channel lengths, number and type of discontinuities, and other factors. Due to the large number of possible system partitioning and interoperability options for multi-segment channels, they will not be covered in this specification. Instead, details may be found in the *PCI Express Card Electromechanical Specification* and other form factor specifications.

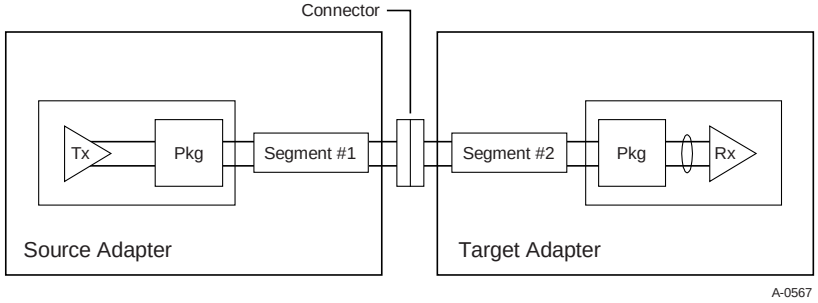


Figure 4-83: Multi-segment Channel Example

4.3.6.3. Upper Limit on Channel Capacitance

The interconnect total capacitance to ground seen by the Receiver Detection circuit must not exceed 3 nF to ground, including capacitance added by attached test instrumentation. This limit is needed to guarantee proper operation during Receiver detect. Note that this capacitance is separate and distinct from the AC coupling capacitance value (see Table 4-18).

4.3.6.4. 8.0 GT/s Channel Compliance Testing

This section of the specification is relevant only for those cases where a platform design comprehends the relevant channel between transmitter device pins and receiver device pins. These types of platform designs are called "captive channels". Designs that are not captive channels shall refer to the appropriate form factor (CEM) specification, since in this case the CEM specification takes precedence over the base specification.

The key components and processes of channel tolerancing are illustrated in Figure 4-84. The basic channel compliance approach is to first acquire the channel's characteristics, usually by means of s-parameters or equivalent model. Behavioral Tx and Rx package models are then appended to the channel model to form a die to die topology. The model shall include both victim and a sufficient number of aggressor paths to accurately capture channel crosstalk effects. Using the Tx voltage and jitter limits defined in the transmitter specification section it is possible to transform these parameters to what would appear at the die pad of a Tx.

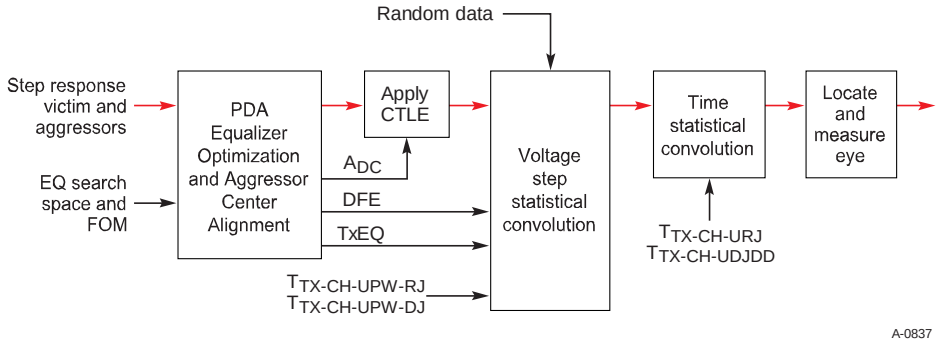


Figure 4-84: Flow Diagram for Channel Tolerancing

The resulting model is analyzed via simulation, yielding voltage and jitter at a point equivalent to the input latch of the receiver. The signal so observed at the receiver's latch is referenced to a recovered data clock from which an eye diagram can be constructed.

The simulation process must also properly account for Tx and Rx equalization optimization as occurs in the minimum capability Tx/Rx pair. This means the simulation process must be able to select the optimum values for the Tx presets or coefficients and Rx equalization settings based upon a 1st order CTLE and a 1-tap DFE.

4.3.6.4.1. Behavioral Transmitter and Receiver Package Models

Tx and Rx behavioral package models are defined in this specification to represent the maximum loss die plus package that are expected to interoperate with the range of channels. The models are included as an appendix to the specification as design collateral. Each model comprehends the C_{PIN} and C_{PAD} parasitic capacitances plus a differential t-line element as illustrated in Figure 4-85. Tx and Rx package models have identical channel lengths and identical C_{PIN} values. Separate Tx and Rx models are included to reflect the smaller C_{PAD} capacitance typical in most receiver implementations.

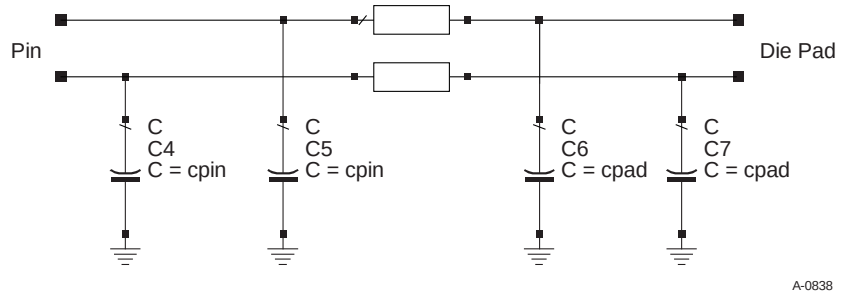


Figure 4-85: Tx/Rx Behavioral Package Models

For ease of incorporation into the post processing flow the behavioral Tx and Rx package models are specified as 4-port s-parameter files. The files are included in an appendix to the base specification with port designations, frequency range and granularity as listed below. The reference impedance for the s-parameters is 50 Ω. File format is Touchstone. Behavioral package model files are available on the PCI-SIG website at: http://www.pcisig.com/specifications/pciexpress/base2/seasim_package/.

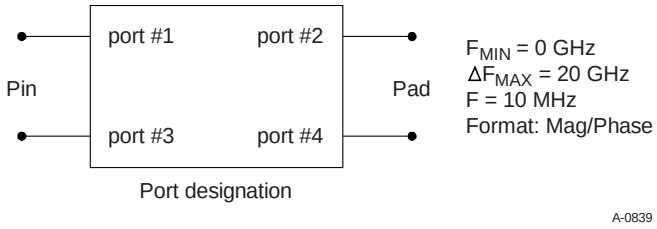


Figure 4-86: Behavioral Tx and Rx S-Parameter File Details

4.3.6.4.2. Simulation Tool Requirements

Channel tolerancing is implemented by means of simulation, where the pass/fail criteria are defined in terms of a time domain eye diagram. The simulation tool must accept a prescribed set of inputs, including the channel under consideration and then simulate based upon a set of post processing requirements. The base specification does not stipulate the use of any specific tool for simulating channels. However any simulation tool must meet the following requirements.

4.3.6.4.3. Simulation Tool Chain Inputs

- **Channel characteristics defined as s-parameters or equivalent model:** The model must include the victim differential lane plus as many aggressors as required to accurately capture crosstalk. In most cases this will be between 2 and 4 additional differential Lanes.
- **Behavioral Tx and Rx package models:** The models will be included as part of the specification in the form of s-parameter files. They may be thought of as a pi network, such as is shown in Figure 4-85.

- **Transmitter Jitter and voltage:** The voltage and jitter parameters input to the simulator may be directly obtained from a combination of the transmitter and Refclk jitter. Since these parameters are fixed the simulation tool may choose to hard code their values.
- **Transmitter and Receiver Termination Impedance:** The simulator shall use a 2 x 50 Ω termination for both the transmitter and receiver. This value matches the assumptions which are implicit in generating and measuring the stressed eye for Rx tolerancing.

4.3.6.4.4. Processing Steps

- **Time domain representation of the end-to-end connectivity:** Included are the behavioral Tx and Rx packages and the channel under test.
- **Tx voltage and jitter:** The voltage and jitter are listed in Table 4-27. They are derived from the voltage and jitter parameters defined for the transmitter, but have been recalculated to properly comprehend high and low frequency jitter components, and also include Refclk jitter contributions.
- **Behavioral TxEq:** The simulator shall replicate the TxEq capabilities defined in the transmitter section and Table 4-19.
- **Behavioral Rx CTLE:** The simulation tool shall implement a behavioral CTLE that replicates the CTLE function employed for Rx tolerancing.
- **Behavioral DFE:** The simulation tool shall implement a 1-tap DFE, where the dynamic range for the feedback coefficient is defined in Section 4.3.4.3.7.
- **Optimizing Tx equalization and Rx DFE/CTLE settings:** The simulation tool shall implement an optimization algorithm that selects the combination of Tx equalization and Rx CTLE and DFE settings that yields a maximum eye width at the far end of the channel. For details refer to Section 4.3.6.5.3.
- **Statistical Treatment of jitter:** In order to avoid overestimating the effect of channel-data and channel-jitter interactions, the tool shall use a statistical analysis of these parameters to generate voltage/jitter eye margins.

4.3.6.4.5. Simulation Tool Outputs

Output eye parameters: The simulator shall generate a statistically defined output that displays the eye width and eye height. EH will be measured as the peak eye height, while EW shall be measured at the zero crossing line. Additionally the simulator shall have the capability to adjust the data sample point by ±0.1 UI from the median center of the UI.

4.3.6.4.6. Open Source Simulation Tool

An open source simulation tool shall be provided with the specification as design collateral. The tool will provide a turnkey capability, where the user provides the channel characteristics at the receiver's die pad as step responses, and the tool calculates a statistical eye showing pass/fail.

4.3.6.5. Behavioral Transmitter Parameters

4.3.6.5.1. Behavioral Tx Jitter and Voltage

Tx voltage and jitter parameters may be computed starting with the definitions and limits defined in Table 4-26. There are some definitional differences between the parameters defined below and their counterparts in Table 4-19. In particular the Tx jitter parameters are defined in terms of DjDD and Tj, while those in Table 4-26 include DjDD and Rj terms. All voltage and jitter components listed in Table 4-26 are defined at the die pad. Note that the combination of the behavioral Tx package plus the worst case voltage/jitter parameters defined below also meet the ps21TX limit defined in Table 4-19.

Table 4-26: Jitter/Voltage Parameters for Channel Tolerancing

Symbol	Parameter	Value	Units	Notes
$V_{TX-CH-FS-NO-EQ}$	Full swing Tx voltage	804	mVPP	Full swing, No Tx Eq. See Note 1.
$V_{TX-CH-RS-NO-EQ}$	Reduced swing Tx voltage	402	mVPP	Reduced swing, No Tx Eq. See Note 1
$T_{TX-CH-URJ}$	Tx uncorrelated Rj	1.55	ps RMS	No DDj of HF jitter. See Note 2.
$T_{TX-CH-UDJDD}$	Tx uncorrelated DjDD	7.0	ps PP	No DDj of HF jitter
$T_{TX-CH-UPW-RJ}$	Uncorrelated PW Rj	1.0	ps RMS	
$T_{TX-CH-UPW-DJ}$	PW DDj	10	ps PP	See Note 3.
$T_{TX-DIEPAD-EDGERATE}$	Signal edge rate at behavioral Tx die pad	35	ps	Measured 10% to 90% using a gaussian lowpass filter to shape the edge. See Note 4.

Notes:

1. Voltages selected to yield 800 mVPP and 400 mVPP respectively at pin of behavioral Tx package when measured at low frequency.
2. Includes low frequency (non F/2) Rj components from the transmitter and Rj from the Refclk.
3. Applied on a per edge basis as a dual Dirac model.
4. Does not include parasitic die pad capacitance. See Figure 4-85 for details of behavioral package.

4.3.6.5.2. Parameter Definitions for Table 4-26 Entries

This section is for informative purposes. The parameters defined in the above table are derived from the transmitter voltage and jitter parameters, but are referenced to the die pad. This is necessary to allow the channel simulation to include a behavioral Tx package and drive the package from the die pads. Additionally, the Tj terms must be decomposed into separate Rj and DjDD terms.

- $V_{TX-CH-FS-NO-EQ}$ and $V_{TX-CH-RS-NO-EQ}$: These two parameters define the minimum peak-peak voltage corresponding to Vd in Figure 4-42.
- The jitter parameters are derived based on the following set of equations. Algebraic manipulation is used to extract the Rj implicitly defined by the combination of Tj and DjDD terms.

jit_hfrj_nui = (T_{TX-UTJ} – $T_{TX-UDJ-DD}$)/14.06 = 1.37ps
$T_{TX-CH-UPW-RJ}$ = ($T_{TX-UPWJ-TJ}$ – $T_{TXUPWJ-DJDD}$)/14.06 = 1.00ps
$T_{TX-CH-UPW-DJ}$ = $T_{TXUPWJ-DJDD}$ = 10.0ps
$T_{TX-CH-URJ}$ = sqrt(jit_hfrj_nui**2 – ($T_{TX-CH-UPW-RJ}$*0.707)**2 + $T_{REFCLK-RMS}$**2) = 1.55 ps
$T_{TX-CH-UDJDD}$ = $T_{TX-UDJ-DD}$ – ($T_{TXUPWJ-DJDD}$)/2 = 7.00ps

A-0840

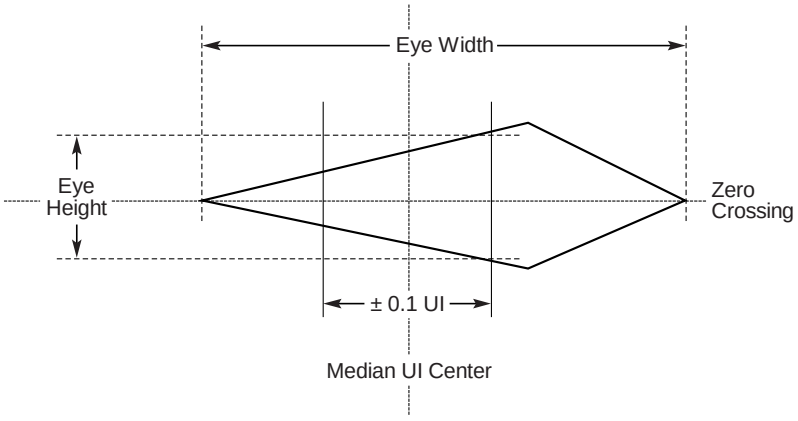
Figure 4-87: Derivation of Jitter Parameters in Table 4-26

4.3.6.5.3. Algorithm for Optimizing Tx/Rx Equalization

The algorithm for optimizing Tx/Rx equalization shall operate as follows. For every combination of CTLE and TxEQ coefficients calculate the zero forced DFE and use PDA to find maximum eye area as figure of merit within the constraints set by the ±0.1 UI aperture and ±30 mV d1 coefficient limits.

4.3.6.5.4. Pass/Fail Eye Characteristics

The output of the simulation tool shall be in the form of pass/fail characteristics as defined by an eye mask as shown in Figure 4-88. EH and EW must meet respectively the voltage and jitter parameters defined in Table 4-27.



A-0841

Figure 4-88: EH, EW Mask

Table 4-27: Channel Tolerancing Eye Mask Values

Symbol	Parameter	Value	Units	Comments
$V_{RX-CH-EH}$	Eye height	25 (min)	mVPP	Eye height at BER=10^{-12}. Note 1.
$T_{RX-CH-EW}$	Eye width at zero crossing	0.3 (min)	UI	Eye width at BER=10^{-12}
$T_{RX-DS-OFFSET}$	Peak EH offset from UI center	±0.1	UI	See Figure 4-88 for details.
V_{RX-DFE_COEFF}	Range for DFE d_1 coefficient	±30	mV	See Figure 4-70 for details.

Notes:

1. $V_{RX-CH-EH}$ is defined as max EH within an aperture of ±0.1 UI from mean UI center. See Figure 4-88 for details.

4.3.6.5.5. Characterizing Channel Common Mode Noise

A channel must meet the common mode requirements as they are defined at the receiver. In general, it is not possible to accurately simulate all the channel's common mode noise contributions due to the large number of mechanisms that can generate CM noise, including the transmitter. Typically channel common mode noise is a budgeted parameter, and the limits defined below assume a budgeting process. The channel's CM limit is defined as the amount of CM noise that a channel can add and still meet the Rx CM limits assuming the worst case Tx CM. This limit is 75 mVPP for EH < 100 mV and 125 mVPP for EH ≥ 100 mVPP.

Note that the Tx and channel CM noise parameters cannot simply be added to obtain the Rx CM limit. This is due to the fact that a channel will attenuate some of high frequency Tx CM noise while propagating Tx LF CM noise through with little loss. The channel may also contribute both high and low frequency CM components of its own.

4.3.6.5.6. Verifying $V_{CH-IDLE-DET-DIFFp-p}$

$V_{CH-IDLE-DET-DIFFp-p}$ is defined to guarantee that, when a transmitter issues an EIEOS sequence, the receiver is guaranteed to detect it. Potentially larger TxEq boost ratios at 8.0 GT/s necessitate that this parameter be verified; this procedure was not necessary for 2.5 or 5.0 GT/s, where the max TxEq boost is smaller. Defining the launch and detect voltages at the Tx/Rx die pad permits $V_{CH-IDLE-DET-DIFF-PP}$ to be verified with the same channel model used to determine eye margins. It is also acceptable, to simulate from Tx pin to Rx pin, (excluding the Tx and Rx behavioral package models), in which case the EIEOS and idle detect parameters defined in the Tx and Rx sections are applicable.

Long channels, where $V_{TX-EIEOS-FS}$ is applicable, are characterized by driving the channel under test with the EIEOS pattern and -11.0 dB de-emphasis and zero dB preshoot. For short channels, where $V_{TX-EIEOS-RS}$ is applicable, -4.5 dB of de-emphasis and zero dB of preshoot are applied.

Table 4-28: EIEOS Signaling Parameters

Parameter	Description	Value	Units	Comments
$V_{CH\text{-}IDLE\text{-}EXIT\text{-}pp}$	Idle detect voltage seen at the Rx die pad	172	mVPP	Assuming Rx R_{TERM} of 2 x 50 Ω
$V_{CH\text{-}EIEOS\text{-}FS\text{-}Vb}$	PP voltage during Vb interval at behavioral Tx die pad for full swing signaling	255	mVPP	Assuming Tx R_S of 2 x 50 Ω
$V_{CH\text{-}EIEOS\text{-}RS\text{-}Vb}$	PP voltage during Vb interval at behavioral Tx die pad for reduced swing signaling	237	mVPP	Assuming Tx R_S of 2 x 50 Ω

4.3.7. Refclk Specifications

4.3.7.1. Reference Clock Specifications for 5.0 GT/s

This section refers to 5.0 GT/s only. Specifications for the Refclk in 2.5 GT/s systems appear in the *PCI Express Card Electromechanical Specification*. Additionally, the analysis of Refclk jitter and the statistical methodology derived to characterize Refclk phase jitter are covered in detail in the JWG white paper.

PCI Express Base Specification, Rev. 1.1 did not include Refclk parameters; instead, parameters for Refclk were defined in *PCI Express Card Electromechanical Specification, Rev. 1.1*. While developing the 5.0 GT/s specification, it became obvious that it made sense to include the Refclk as part of the electrical specification, and so it is included in this specification.

4.3.7.1.1. Spread Spectrum Clock (SSC) Sources

The data rate may be modulated from +0% to -0.5% of the nominal data rate frequency, at a modulation rate in the range not exceeding 30 kHz – 33 kHz. The ±300 ppm requirement still holds, which requires the two communicating Ports be modulated such that they never exceed a total of 600 ppm difference. For most implementations this places the requirement that both Ports require the same bit rate clock source when the data is modulated with an SSC.

4.3.7.1.2. Refclk Architectures for 5.0 GT/s

Three distinct Refclk architectures are possible: common Refclk, separate Refclks, and data driving PLL. Each has an associated filter function that comprehends the worst case combination of PLL bandwidth/peaking and equivalent jitter. The effective jitter seen at the Rx's clock-data recovery inputs is a function of the difference in Rx and Tx PLL bandwidth and peaking convolved with the jitter spectrum of the Refclk. It is also dependent on the Refclk architecture.

4.3.7.2. Refclk Test Setup

The test setup for the Refclk assumes that only the Refclk generator itself is present. Provision is made in the test setup to account for signal degradation that occurs between the pins of the Refclk

generator and the Transmitter or Receiver in an actual system. The above described setup emulates the worst case signal degradation that is likely to occur at the pins of a PCI Express device.

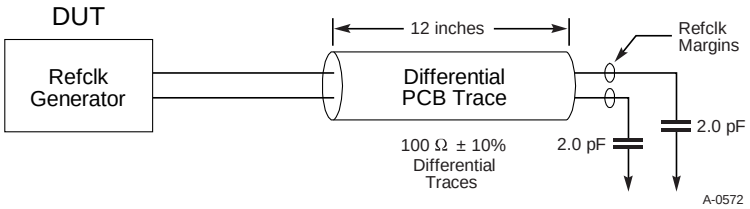


Figure 4-89: Refclk Test Setup

4.3.7.3. Bit Rate Tolerance and Spread Spectrum Clocking

The tolerance for the Refclk is 100 MHz ±300 ppm, where this number is defined with spread spectrum clocking (SSC) turned off. SSC may be implemented as a method of reducing EMI. The Refclk rate for a single source may be modulated from +0 to -5000 ppm of the nominal data rate frequency, at a modulation rate lying within a 30 kHz – 33 kHz range. The maximum rate of change of the frequency on a reference clock with SSC active is 1250 ppm/µs. Refclks that operate with SSC need to meet an additional frequency dependent phase jitter limit at low frequency. The data points in Table 4-29 are used to construct a piece-wise linear graph from 30 kHz to 500 kHz and the unfiltered phase jitter of the reference clock in the frequency domain must fall below this graph as shown in Figure 4-90.

Comment [VS154]: ECN: SRIS

Table 4-29: Limits for phase jitter from the reference clock

Frequency	Maximum Peak to Peak Phase Jitter (ps)
30 kHz - 33 kHz	25000
100 kHz	1000
500 kHz	25

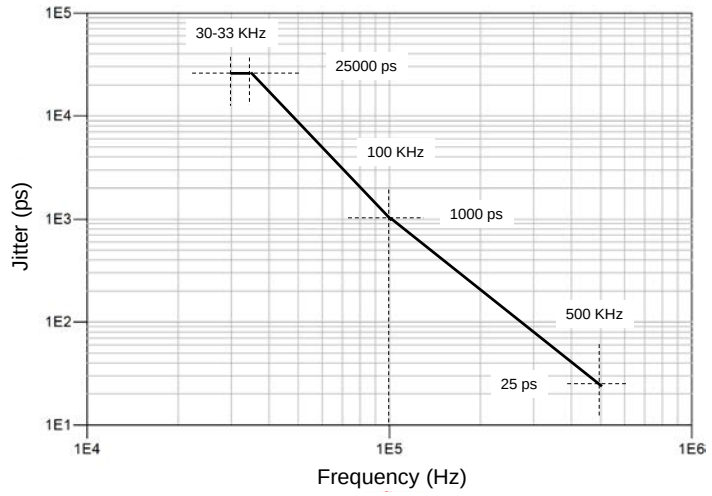


Figure 4-90: Limits for phase jitter from the Reference Clock

Comment [VS155]: D-ECN___SRIS_JTOL_SSC__31_March_14

4.3.7.3.1. Filtering Functions Applied to Refclk Measurements

Raw Refclk data contains jitter over a wide range of frequencies, some of which will be tracked by the Receiver or otherwise removed by the combination of the Tx and Rx PLLs. Consequently, it is necessary to apply a series of filter operations to raw Refclk data in order to obtain meaningful jitter measurements. The nature of the filter functions is in part dependent on the Refclk architecture. For example, the PLL functions for the common Refclk Rx and data clocked Rx architectures are different and therefore yield differing amounts of HF Rj. Table 4-30 lists the filter functions for common Refclk Rx and data clocked Rx architectures. In general, there are five different filter functions applied:

- SSC separation: Used to remove SSC components from the low frequency range allowing one to define separate low frequency Rj and low frequency Dj components
- 0.01 – 1.5 MHz step BPF: Lower edge of filter removes 1/f jitter contributions that are completely trackable by CDR. This function also removes high frequency jitter components.
- 1.5 MHz step HPF: Removes low frequency jitter components, allowing one to define those jitter components not trackable by CDR.
- Edge filtering: Minimizes test equipment measurement artifacts caused by finite sampling/voltage resolution aperture. This is a voltage averaging process that is applied at a frequency of 5 GHz.
- PLL difference function or Max PLL BW function. The first is applied to the common Refclk Rx architecture and comprehends worst case mismatch between Tx and Rx PLLs and impact of transport delay. The second is applied to data clocked Rx architecture and comprehends maximum PLL BW and peaking.

Table 4-30: Filtering Functions Applied to Refclk Measurements

Refclk Architecture	Common Refclk Rx	Data Clocked Rx
< 1.5 MHz jitter components	SSC separation PLL difference function 0.01 - 1.5 MHz step BPF	No SSC separation Max PLL BW function 0.01 - 1.5 MHz step BPF
> 1.5 MHz jitter components	PLL difference function 1.5 MHz step HPF Edge filtering	Max PLL BW function 1.5 MHz step HPF Edge filtering

Implementation details for the PLL difference and maximum PLL functions are described in the sections relating to common Refclk Rx and data clocked Rx architectures that follow. The 1.5 MHz HPF and 0.01 – 1.5 MHz BPF characteristics are described in Figure 4-49.

4.3.7.3.2. Common Refclk Rx Architecture

This architecture implies that a single Refclk source is distributed to the Tx and Rx. As a consequence, much of the SSC jitter sourced by the Refclk is propagated equally through Tx and Rx PLLs, and so intrinsically tracks. Figure 4-91 illustrates the common Refclk Rx architecture, showing key noise, delay, and PLL transform sources. Since Refclk is common to both Tx and Rx, its noise characteristic, $X_1(s)$ is driven to both Tx and Rx PLLs. The amount of jitter appearing at the CDR is then defined by the difference function between the Tx and Rx PLLs, HCC(s).

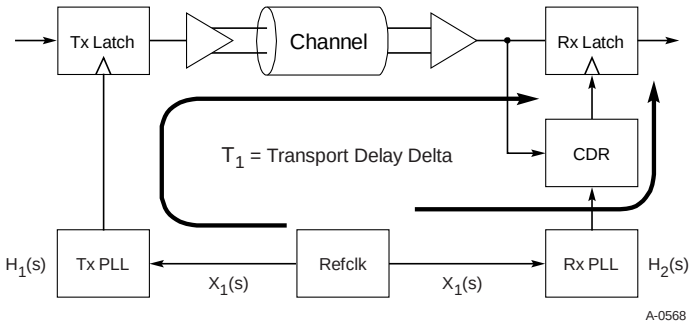


Figure 4-91: Common Refclk Rx Architecture

Based on the above clock architecture, it is possible to define a difference function that corresponds to the worst case mismatch between Tx and Rx PLLs. Second order transfer functions are assumed, even though most PLL transfer functions are 3rd order or higher, since 2nd order functions tend to yield a slightly conservative difference function vis-a-vis an actual PLL.

$$X_{CC}(s) = X_1(s) * H_{CC}(s)$$

$$H_{CC}(s) = \left[\frac{2s\zeta_1\omega_{n1} + \omega_{n1}^2}{s^2 + 2s\zeta_1\omega_{n1} + \omega_{n1}^2} e^{-sT_1} - \frac{2s\zeta_2\omega_{n2} + \omega_{n2}^2}{s^2 + 2s\zeta_2\omega_{n2} + \omega_{n2}^2} \right] \quad \textbf{Equation 4.3.4}$$

Jitter contribution from H_1 Jitter contribution from H_2

Conversion between the natural PLL frequency ω_n and the -3 dB point is given by the following expression.

$$\omega_{3dB} = \omega_n \sqrt{1 + 2\zeta^2 + \sqrt{(1 + 2\zeta^2)^2 + 1}} \quad \textbf{Equation 4.3.5}$$

In the common Refclk Rx architecture it is also necessary to comprehend a maximum Transmitter to Receiver transport delay difference. This delay delta is illustrated in Figure 4-92 and represents the delay difference between the Transmitter data and recovered Receiver clock as seen at the inputs to the receive latch.

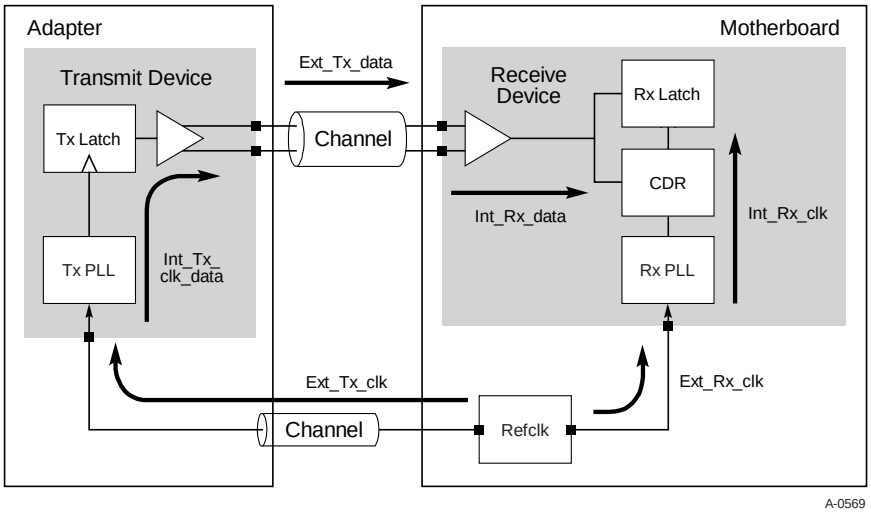


Figure 4-92: Refclk Transport Delay Paths for a Common Refclk Rx Architecture

PLL BW and peaking numbers are defined in Table 4-31. The filter function is defined by $H_{CC}(s)$ and includes expressions for the min and max PLL characteristics, where the smaller PLL BW corresponds to ω_{n1} and ζ_1 and the larger PLL BW corresponds to ω_{n2} and ζ_2. The e^{-sT1} term corresponds to the transport delay delta as illustrated in Figure 4-92.

Table 4-31: Difference Function Parameters Applied to Refclk Measurement

Symbol	Parameter	Min	Max	Units	Comments
T_1	Data/clock transport delay delta		12	ns	See Note 1.
ω_1	PLL #1 natural frequency	$4.31*2\pi$ or $1.82*2\pi$		Mrad/s	See Notes 1, 2, and 3.
ζ_1	PLL #1 damping factor	0.54 or 1.16	1.75 (0.5 dB)		See Notes 1 and 2.
ω_2	PLL #2 natural frequency		$8.61*2\pi$	Mrad/s	See Note 1.
ζ_2	PLL #2 damping factor	0.54 or 1.16	1.75 (0.5 dB)		See Notes 1, 2, and 4.

Notes:

1. T1 defines to the cumulative transport delay delta of the data and Refclk paths as shown in Figure 4-92 and includes both off-chip and on-chip delay terms. Maximum internal transport delay for Tx and Rx is 2.0 ns.
2. For the common Refclk Rx architecture, two possible combinations of minimum PLL BW and corresponding peaking are specified. If the min PLL BW is ≥5 MHz, then a max peaking of 1.0 dB (corresponding to $\zeta = 1.16$) is required. If the min PLL BW is ≥8 MHz, then 3 dB of peaking (corresponding to $\zeta = 0.54$) is allowed.
3. The natural frequency limits for PLL #1 correspond to -3 dB cutoff frequencies of 8.0 MHz ($4.31e6*2\pi$) and 5.0 MHz ($1.82e6*2\pi$).
4. The natural frequency limit for PLL #2 of $8.61e6*2\pi$ corresponds to a -3 dB cutoff frequency of 16 MHz.

4.3.7.3.3. Compliance Parameters for Common Refclk Rx Architecture

Table 4-32 defines the compliance parameters for the common Refclk Rx architecture.

Table 4-32: Refclk Parameters for Common Refclk Rx Architecture at 5.0 GT/s

Symbol	Description	Limits		Units	Note
		Min	Max		
$T_{REFCLK-HF-RMS}$	> 1.5 MHz to Nyquist RMS jitter after applying Equation 4.3.3		3.1	ps RMS	1
$T_{REFCLK-SSC-RES}$	SSC residual		75	ps	1
$T_{REFCLK-LF-RMS}$	10 kHz - 1.5 MHz RMS jitter		3.0	ps RMS	2
$T_{SSC-FREQ-DEVIATION}$	SSC deviation		+0.0/-0.5	%	
$T_{SSC-MAX-PERIOD-SLEW}$	Maximum SSC df/dt		0.75	ps/UI	3

Notes:

1. $T_{REFCLK-HF-RMS}$ is measured at the far end of the test circuit illustrated in Figure 4-89 after the filter function defined in Table 4-30 for Common Refclk Rx for >1.5 MHz jitter components has been applied.
2. $T_{REFCLK-SSC-RES}$ and $T_{REFCLK-LF-RMS}$ are measured after the filter function defined in Table 4-30 for Common Refclk Rx for >1.5 MHz jitter components has been applied.
3. Defined for a worst case SSC modulation profile such as Lexmark.

4.3.7.3.4. Data Clocked Rx Architecture

This architecture does not use Refclk at the Receiver during data recovery, but instead uses the embedded clock edge information contained in the data to directly drive the Rx CDR as shown in Figure 4-93.

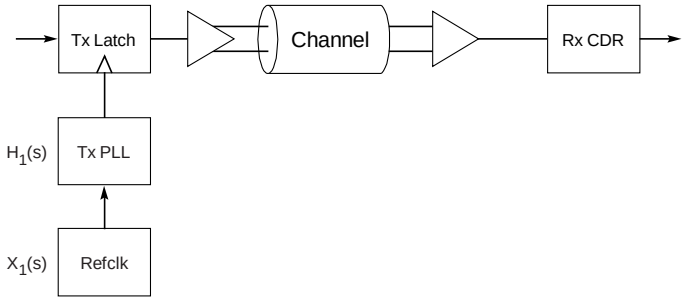


Figure 4-93: Data Clocked Rx Architecture

The difference function for the above data clocked Rx architecture is defined in the following equation:

$$X_{DC}(s) = X_1(s) * H_1(s)$$

$$H_1(s) = \left[\frac{2s\zeta_1\omega_{n1} + \omega_{n1}^2}{s^2 + 2s\zeta_1\omega_{n1} + \omega_{n1}^2} \right]$$

Equation 4.3.6

In the data clocked Rx architecture, the amount of Refclk jitter propagated depends on the maximum PLL bandwidth (16 MHz, 3 dB of peaking). It is also the case that the Rx CDR must track the entirety of SSC, (20 ns) and the CDR must be capable of tracking SSC at a maximum slew rate corresponding to the largest df/dt SSC modulation profiles.

Table 4-33: PLL Parameters for Data Clocked Rx Architecture

Symbol	Parameter	Min	Max	Units	Comments
ω_1	Tx PLL natural frequency		8.61*2π	Mrad/s	See Note 1.
ζ_1	Tx PLL damping factor	0.54 (3.0 dB)	1.75 (0.5 dB)		See Notes 1 and 2.

Notes:

1. The ω_1 and ζ_1 correspond to 16 MHz with 3.0 dB of peaking. Note that for the data driving architecture we cannot take advantage of the differencing function for two PLLs and must instead apply the full 0-16 MHz/3.0 dB peaking PLL transfer function. Similarly, the lack of an Rx PLL obviates the need for defining a transport delay parameter.
2. A minimum peaking is also specified in order to place an upper limit on the amount of energy in the rolloff of the PLL. Since ζ_1 defines both the peaking and rolloff, a minimum and maximum for ζ_1 uniquely defines the amount of BW in the rolloff region.

Formatted: Subscript

Formatted: Subscript

4.3.7.3.5. Compliance Parameters for Data Clocked Rx Architecture

Table 4-34 defines the Refclk jitter parameters for a Refclk in a data clocked Rx architecture.

Table 4-34: Refclk Parameters for Data Clocked Rx Architecture

Symbol	Description	Limits		Units	Note
		Min	Max		
$T_{REFCLK-HF-RMS}$	1.5 - Nyquist MHz RMS jitter after applying Equation 4.3.5		4.0	ps RMS	1
$T_{REFCLK-SSC-FULL}$	Full SSC modulation corresponding to +0 – 0.5%		20	ns	1
$T_{REFCLK-LF-RMS}$	10 kHz - 1.5 MHz RMS jitter		7.5	ps RMS	2
$T_{SSC-FREQ-DEVIATION}$	SSC deviation		+0.0/-0.5	%	
$T_{SSC-MAX-PERIOD-SLEW}$	Max SSC df/dt		0.75	ps/UI	3

Notes:

1. $T_{REFCLK-HF-RMS}$ is measured at the far end of the test circuit illustrated in Figure 4-93 after the filter function defined in Table 4-30 for Data Clocked Rx for >1.5 MHz jitter components has been applied.
2. $T_{REFCLK-SSC-FULL}$ and $T_{REFCLK-LF-RMS}$ are measured after the filter function defined in Table 4-30 for Data Clocked Rx for < 1.5 MHz jitter components has been applied.
3. Defined for a worst case SSC modulation profile such as Lexmark.

4.3.7.4. Separate Refclk with No SSC (SRNS) Architecture

Comment [VS156]: ECN:SRIS

It is also possible to architect a PCI Express implementation with separate Refclk sources for the Tx and Rx. Since this architecture employs two independent clock sources, the amount of jitter impinging on the Receiver is the RSS sum, rather than the difference of the PLL transfer characteristics. As a consequence, the jitter requirements for the Refclks in a separate Refclk

architecture are substantially tighter than for a common Refclk Rx architecture. Furthermore, it is not in general possible to guarantee interoperability between separate Refclk architecture components and those using other clock architectures. For example, a separate Refclk adapter will not interoperate with a Root Complex driving data with SSC. ~~For this reason, t~~This specification does not explicitly define the requirements for separate Refclk architecture, but instead will defer to the appropriate form factor specification.

When tolerancing a PCI Express Rx in ~~a~~ this separate Refclk architecture, the generator and DUT are furnished with separate Refclks that each are within ±300 PPM of the nominal frequency of 100 MHz. SSC must be turned off for both Refclk sources. These limitations are required to guarantee proper operation of the buffering and flow control in the Rx.

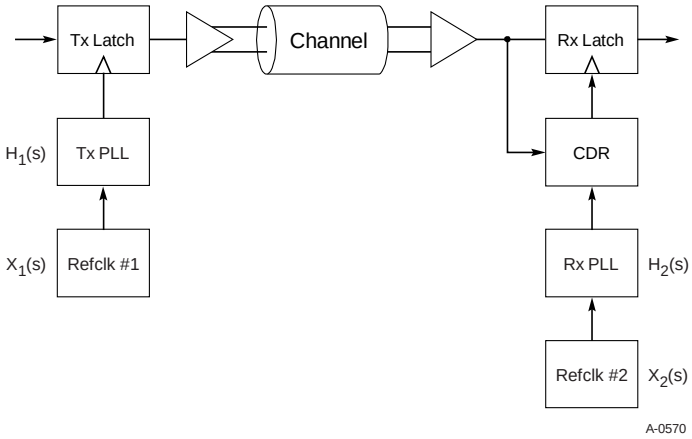


Figure 4-94: Separate Refclk With No SSC (SRNS) Architecture

The impact of tighter Refclk margins for the separate Refclk architecture may be seen by examining the phase jitter transfer equations below. Since both Refclks are independent, their phase jitter is passed through the Tx and Rx PLLs independently, and the maximum PLL BW/peaking of 16 MHz/3 dB must be assumed. Additionally, since both Tx and Rx Refclks are independent, their Rj terms add as an RSS value. Consequently, the jitter characteristics for the separate Refclk architecture must be considerably tighter than for the other two architectures.

$$H_1(s) = \left[\frac{2s\zeta_1\omega_{n1} + \omega_{n1}^2}{s^2 + 2s\zeta_1\omega_{n1} + \omega_{n1}^2}\right] \quad H_2(s) = \left[\frac{2s\zeta_2\omega_{n2} + \omega_{n2}^2}{s^2 + 2s\zeta_2\omega_{n2} + \omega_{n2}^2}\right]$$

Equation 4.3.7

$$X_{SC}(s) = \sqrt{\left[X_1(s) * H_1(s)\right]^2 + \left[X_2(s) * H_2(s)\right]^2}$$

Comment [VS157]: ECN: SRIS

Comment [VS158]: ECN: SRIS

4.3.7.5. Separate Refclk with Independent SSC (SRIS) Architecture

Comment [VS159]: ECN:SRIS

It is also possible to architect a PCI Express implementation with separate Refclk sources with independent SSC for the Tx and Rx. This architecture employs two independent clock sources each having independent spread spectrum clocking (SSC). The Receiver must be able to track and reject the resultant phase drift between the Tx and Rx. Changes in the logical sub-block are also required for this separate Refclk with Independent SSC architecture; refer to the logical sub-block section for all relevant information concerning SRIS (Sections 4.2.5.6, 4.2.7, 4.2.7.3 and 4.2.7.4).

Figure 4-95 below shows the block diagram of the separate Refclk architecture. Since Refclk #1 and Refclk #2 are independent from each other, their respective jitter at the Rx Latch are simply $X_1(s)*H_1(s)*H_CDR(s)$ and $X_2(s)*H_2(s)*H_CDR(s)$; where H_CDR(s) is the CDR's jitter transfer function. However, since Refclk jitter is primarily Rj, their combined impact should be the Root Sum Square (RSS) of the individual terms.

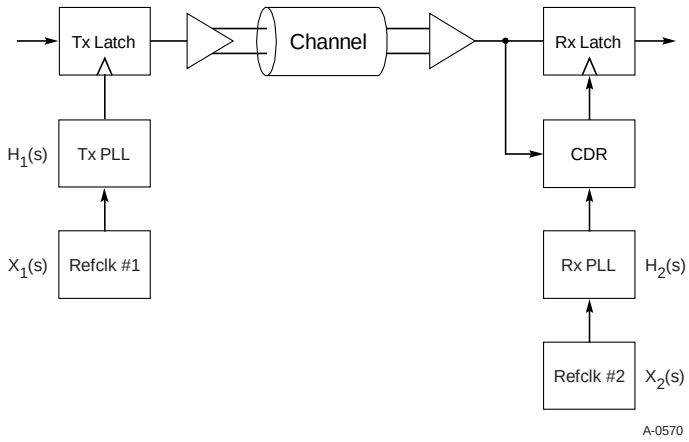


Figure 4-95: Separate Refclk With Independent SSC Architecture

For the separate Refclk with independent SSC architecture, the informative CDR jitter transfer function is a second order high-pass function with the -3 dB corner frequency at 5 MHz, see Figure 4-96 and Equation 4.3.7.

Formatted: Font color: Black

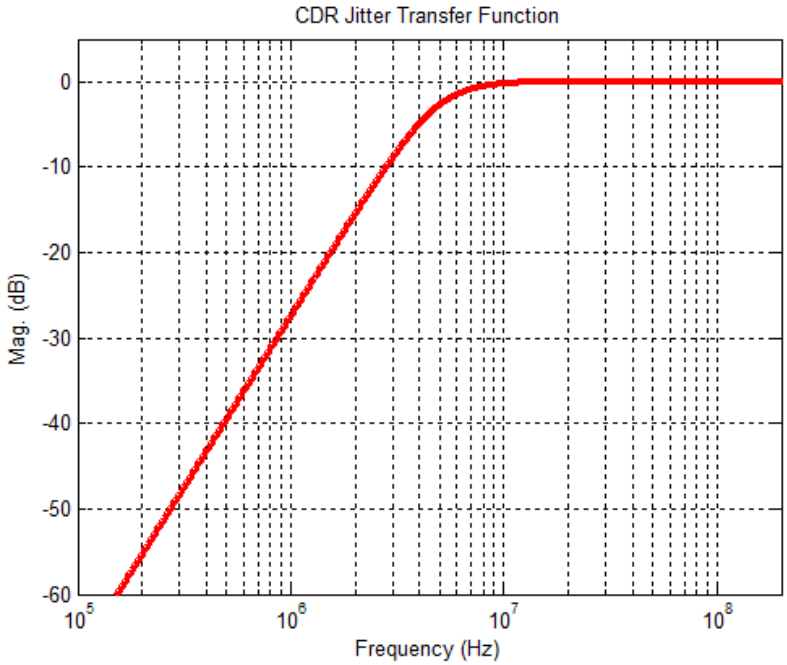


Figure 4-96: Informative CDR Jitter Transfer Function for the Separate Refclk with Independent SSC (SRIS) Architecture at 5.0 GT/s

$$H_{JTF}(s) = \frac{s^2}{s^2+2\zeta\omega_m s+{\omega_m}^2}$$ Equation 4.3.8

where $f_m = \frac{2*5.0\ \text{MHz}}{\sqrt{1+2\zeta^2+\sqrt{1+(1+2\zeta^2)^2}}}$ $\zeta = .707$ and $\omega_m = 2\pi f_m$.

Table 4-35: Parameters for Separate Refclk With Independent SSC Architecture (SRIS) at 5.0 GT/s

Symbol	Description	Limits		Units
		Min	Max	
F_{REFCLK}	Refclk frequency	99.97	100.03	MHz
$T_{REFCLK-RMS-SRIS}$	RMS Refclk jitter for Separate Refclk Independent SSC architecture[1]		2.0	ps RMS
F_{SSC}	SSC frequency range	30	33	kHz
$T_{SSC-FREQ-DEVIATION}$	SSC deviation		+0.0/-0.5	%

Notes:

1. The Refclk jitter is measured after applying PLL transfer function (16 MHz, 2 dB) and CDR jitter transfer function defined in Figure 4-96.
2. The maximum rate of change of the clock frequency is 1250 ppm/µs.

Table 4-36: Informative Total Jitter Budget for 5.0 GT/s Signaling for Separate Refclk With Independent SSC (SRIS) Architecture

Jitter Contribution	Max Dj (ps)	Tj at BER 1e-12 (ps)
Tx + Refclk #1	30	64.5
Media	58	58
Rx + Refclk #2	63.2	97.7

To summarize, a PCI Express link at 5.0 GT/s can be built using the separate Refclk with independent SSC architecture. Relative to the common Refclk architecture, the only electrical specification changes at 5.0 GT/s are the receiver CDR's jitter transfer function (Figure 4-96), the resulting Refclk jitter number (Table 4-35), and the requirement to test the receiver with separate reference clocks with independent SSC enabled.

4.3.8. Refclk Specifications for 8.0 GT/s

4.3.8.1. CDR Tracking Function

When Refclk data is captured as a raw v(t) vs. time record, it contains jitter components, some of which are tracked by the receiver in an actual PCI Express implementation. It is necessary to remove these low frequency jitter components to reflect the amount of Refclk jitter that actually impinges upon the Rx sampler after passing through the Rx CDR.

Two types of CDRs are defined: a first order for the common Refclk Rx architecture and a second order for the data clocked Rx architecture. Figure 4-67 illustrates the behavioral 1st order CDR

characteristics showing 20 dB/decade tracking capability and the $H_3(s)$ transfer function shown in Figure 4-97. A 2nd order CDR is characterized by the transfer function $H_3(s)$ defined in Figure 4-98. The 2nd order CDR function is characterized by both a bandwidth and a peaking range.

4.3.8.1.1. Edge Filtering

Edge filtering is a technique that minimizes the jitter introduced by the finite sampling time of test equipment. Typically, a smoothing function is applied to the discretely sampled data so the zero differential volts can be more accurately determined.

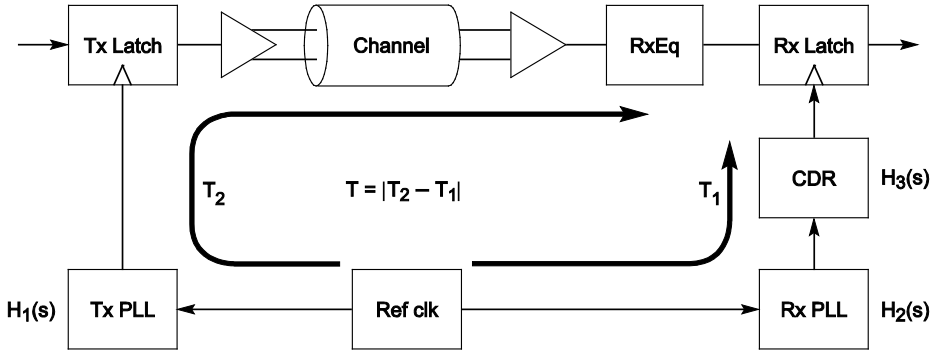
4.3.8.1.2. Tx and Rx PLL Assumptions

Determining the amount of effective PLL jitter seen at the receiver requires that the Refclk transfer function be evaluated. The transfer function differs depending on the Rx architecture. The transfer function for the common Refclk Rx architecture is shown in Figure 4-97, and that for the data clocked architecture in Figure 4-98. Two possible PLL bandwidth and peaking ranges are assumed: a bandwidth of 2-4 MHz with 0.01 – 2.0 dB of peaking and a bandwidth of 2-5 MHz with 0.01 -1.0 dB of peaking. For convenience, the corresponding ω_n and ζ values are defined for the corner limits of PLL BW and peaking.

4.3.8.2. Common Refclk Rx Architecture

Common Refclk Rx architectures are characterized by the Tx and Rx sharing the same Refclk source. The jitter seen by the Rx CDR is a function of the difference between the Tx and Rx PLL transfer functions, with a transport delay term (e^{-sT}), multiplied by the CDR's transfer function. CDR tracking is captured by the $H_3(s)$ term.

Note that the minimum PLL peaking is specified at 0.01 dB while the limit for Tx and Rx PLLs is 0 dB. This small disparity yields a only a small difference in H(s) while still allowing the difference function to be evaluated in a convenient mathematical form.



$$H_1(s) = \left[\frac{2s\zeta_1\omega_{n1} + \omega_{n1}^2}{s^2 + 2s\zeta_1\omega_{n1} + \omega_{n1}^2}\right] \quad H_2(s) = \left[\frac{2s\zeta_2\omega_{n2} + \omega_{n2}^2}{s^2 + 2s\zeta_2\omega_{n2} + \omega_{n2}^2}\right] \quad H_3(s) = \frac{s}{s + \omega_3}$$

$$\left.\begin{aligned} H(s) &= [H_1(s)e^{-sT} - H_2(s)]H_3(s) \text{ or} \\ H'(s) &= [H_2(s)e^{-sT} - H_1(s)]H_3(s) \end{aligned}\right\} \text{Need to compute both}$$

	0.01 dB Peaking	2.0 dB Peaking
$BW_{PLL}(min)$ = 2.0 MHz	ω_{n1} = 0.448 Mrad/s ζ_1 = 14	ω_{n1} = 6.02 Mrad/s ζ_1 = 0.73
$BW_{PLL}(max)$ = 4.0 MHz	ω_{n1} = 0.896 Mrad/s ζ_1 = 14	ω_{n1} = 12.04 Mrad/s ζ_1 = 0.73

	0.01 dB Peaking	1.0 dB Peaking
$BW_{PLL}(min)$ = 2.0 MHz	ω_{n2} = 0.448 Mrad/s ζ_2 = 14	ω_{n2} = 4.62 Mrad/s ζ_2 = 1.15
$BW_{PLL}(max)$ = 5.0 MHz	ω_{n2} = 1.12 Mrad/s ζ_2 = 14	ω_{n2} = 11.53 Mrad/s ζ_2 = 1.15

A-0842

Figure 4-97: 8.0 GT/s Common Refclk Rx Architecture with ω_n, ζ Limits

When characterizing $T_{REFCLK\text{-}RMS\text{-}CC}$ it is recommended that all unique combinations of bandwidth and peaking be included when computing H(s) and H'(s).

Table 4-37: Parameters for Common Refclk Rx Architecture at 8.0 GT/s

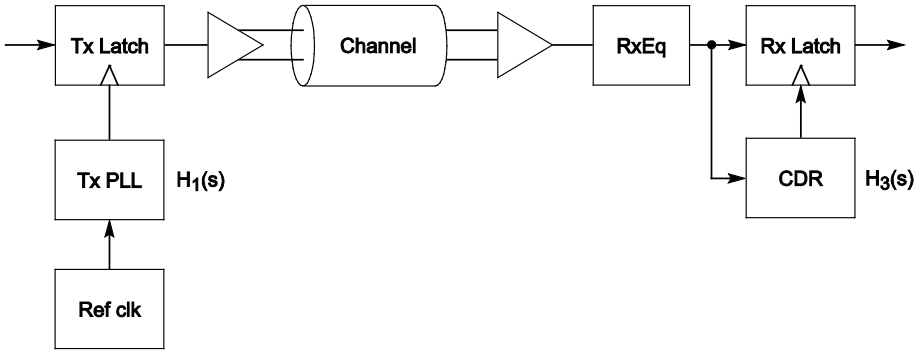
Symbol	Description	Limits		Units
		Min	Max	
F_{REFCLK}	Refclk frequency[1]	99.97	100.03	MHz
$T_{REFCLK-RMS-CC}$	RMS Refclk jitter for common Refclk Rx architecture[3]		1.0	ps RMS
F_{SSC}	SSC frequency range	30	33	kHz
$T_{SSC-FREQ-DEVIATION}$	SSC deviation[2]		+0.0/-0.5	%
$T_{TRANSPORT-DELAY}$	Tx-Rx transport delay	12		ns

Notes:

1. Before application of SSC.
2. It is sufficient to define SSC deviation only without specifying anything about the shape of the modulation envelope. Envelopes with very large df/dt will fail the $T_{REFCLK-RMS-CC}$ parameter.
3. The Refclk jitter is measured after applying the jitter filtering function defined in Figure 4-97.

4.3.8.3. Data Clocked Refclk Rx Architecture

A data clocked Rx architecture is characterized by requiring the receiver's CDR to track the entirety of the low frequency jitter, including SSC. Since the Tx and Rx do not share a common Refclk, the jitter transfer function includes only the Tx PLL transfer function plus the lowpass characteristics of the CDR. Two possible combinations of Tx PLL bandwidth and frequency are defined, and it is recommended that all unique combinations of PLL bandwidth and peaking be considered when computing H(s).



$$H_1(s) = \left[\frac{2s\zeta_1\omega_{n1} + \omega_{n1}^2}{s^2 + 2s\zeta_1\omega_{n1} + \omega_{n1}^2}\right] \qquad H_3(s) = \left[\frac{2s\zeta_3\omega_{n3} + \omega_{n3}^2}{s^2 + 2s\zeta_3\omega_{n3} + \omega_{n3}^2}\right]$$

$$H(s) = H_1(s)[1 - H_3(s)]$$

	0.01 dB Peaking	2.0 dB Peaking
BW_{PLL}(min) = 2.0 MHz	ω_{n1} = 0.448 Mrad/s ζ_1 = 14	ω_{n1} = 6.02 Mrad/s ζ_1 = 0.73
BW_{PLL}(max) = 4.0 MHz	ω_{n1} = 0.896 Mrad/s ζ_1 = 14	ω_{n1} = 12.04 Mrad/s ζ_1 = 0.73

	0.01 dB Peaking	1.0 dB Peaking
BW_{PLL}(min) = 2.0 MHz	ω_{n1} = 0.448 Mrad/s ζ_1 = 14	ω_{n1} = 4.62 Mrad/s ζ_1 = 1.15
BW_{PLL}(max) = 5.0 MHz	ω_{n1} = 1.12 Mrad/s ζ_1 = 14	ω_{n1} = 11.53 Mrad/s ζ_1 = 1.15

	0.5 dB Peaking	2.0 dB Peaking
BW_{CDR}(min) = 10 MHz	ω_{n3} = 16.57 Mrad/s ζ_3 = 1.75	ω_{n3} = 33.8 Mrad/s ζ_3 = 0.73

A-0843

Figure 4-98: 8.0 GT/s Data Clocked Rx Architecture with ω_n, ζ Limits

The minimum PLL peaking is specified at 0.01 dB, while the limit for Tx and Rx PLLs is 0 dB. The difference is very small, and permits H(s) to be evaluated in a convenient mathematical form. When computing the worst case $T_{REFCLK\text{-}RMS\text{-}DC}$ it is necessary to do so for each unique combination of PLL peaking and bandwidth.

BW_{CDR} is defined for the CDR when it is receiving a PRBS pattern. This stipulation is necessary because the value of BW_{CDR} will vary as a function of the transition density of the received data, and the PRBS pattern generates a transition on average every two UI.

Table 4-38: Parameters for Data Clocked Rx Architecture at 8.0 GT/s

Symbol	Description	Limits		Units
		Min	Max	
F_{REFCLK}	Refclk frequency	99.97	100.03	MHz
$T_{REFCLK-RMS-DC}$	RMS Refclk jitter for data clocked Rx architecture[1]		1.0	ps RMS
F_{SSC}	SSC frequency range	30	33	kHz
$T_{SSC-FREQ-DEVIATION}$	SSC deviation		+0.0/-0.5	%

Notes:

1. The Refclk jitter is measured after applying jitter filtering function defined in Figure 4-98.

4.3.8.4. Separate Refclk with no SSC (SRNS) Architecture

Comment [VS160]: ECN: SRIS

The 8.0 GT/s PCI Express Base Specification does not explicitly define the requirements for a separate Refclk architecture. This is the identical position taken by the base specification for 2.5 GT/s and 5.0 GT/s.

When tolerancing a PCI Express Rx in ~~a~~ this separate Refclk architecture, the generator and DUT are furnished with separate Refclks that each are within ±300 PPM of the nominal frequency of 100 MHz. SSC must be turned off for both Refclk sources. These limitations are required to guarantee proper operation of the buffering and flow control in the Rx.

4.3.8.5. Separate Refclk With Independent SSC (SRIS) Architecture

Comment [VS161]: ECN:SRIS

It is possible to architect an 8.0 GT/s PCI Express implementation with separate Refclk sources with independent SSC for the Tx and Rx. Figure 4-95 shows a high level block diagram of the separate Refclk architecture. Since this architecture employs two independent clock sources, the Receiver must be able to track and reject the phase drift due to the two independent SSC clocks. Changes in the logical sub-block are also required for the separate Refclk architecture; please refer to the logical sub-block section for all relevant information concerning SRIS (see Sections 4.2.5.6, 4.2.7, 4.2.7.3 and 4.2.7.4).

For the common Refclk architecture, the 8.0 GT/s specification contains a behavioral CDR model shown in Figure 4-67; it is a first order high-pass function with the -3dB corner frequency at 10MHz. Furthermore, during the receiver stressed jitter test (Section 4.3.4.4), the receiver's CDR bandwidth and its low frequency jitter tracking capability are required to pass the jitter mask (Figure 4-74). For the separate Refclk with independent SSC architecture, the CDR needs to have a second order high-pass behavior to reject the extra phase drift due to independent SSC. This new behavioral CDR jitter transfer function is shown in Figure 4-99 and Equation 4.3.8. The stressed jitter test is also modified to account for the requirement that the CDR can reject jitter from two separate reference clocks with SSC. Other test parameters and requirements such as transmitter jitters (Table 4-19) and the receive stressed voltage eye (Table 4-22) remain unchanged.

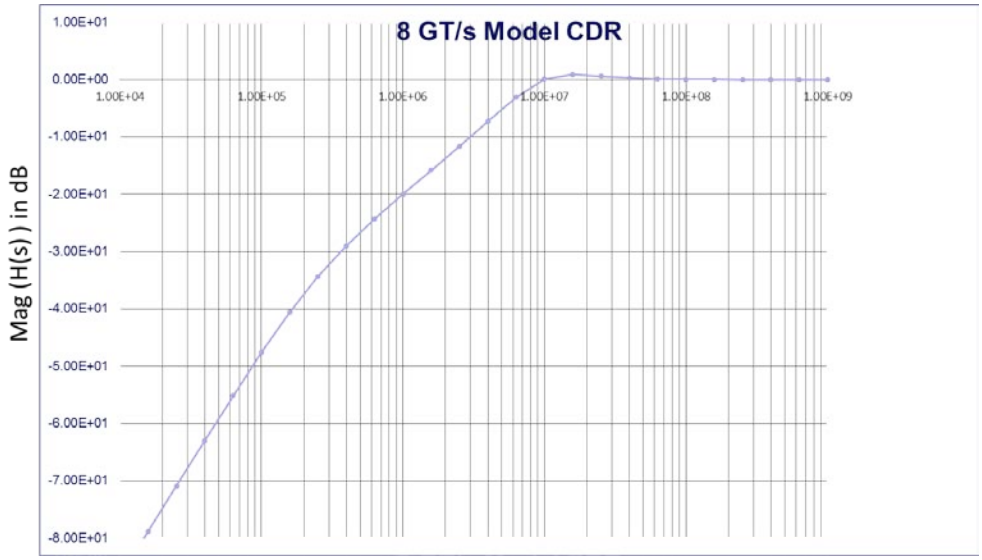


Figure 4-99: Informative CDR Jitter Transfer Function for the Separate Refclk With Independent SSC (SRIS) Architecture at 8.0 GT/s

$$H(s) = \frac{s^2}{s^2+sA+B} \cdot \frac{s^2+2\zeta_2\omega_0 s+\omega_0^2}{s^2+2\zeta_1\omega_0 s+\omega_0^2} \qquad \text{Equation 4.3.9}$$

where

$\zeta_1 = \frac{1}{\sqrt{2}}; \zeta_2 = 1;\ \omega_0 = 10^7 \times 2\pi$

$A = 10^7 \times 2\pi; B = 2.2 \times 10^{12} \times (2\pi)^2$

The new CDR jitter transfer function impacts the Refclk jitter specification. This Refclk jitter number is captured as $T_{REFCLK\text{-}RMS\text{-}SRIS}$ in Table 4-39.

Comment [VS162]: D-ECN___SRIS_JTOL_SSC__31_March_14

Comment [VS163]: 3.1a Update: Added missing exponent 2

Table 4-39: Parameters for Separate Refclk With Independent SSC (SRIS) Architecture at 8.0 GT/s

Symbol	Description	Limits		Units
		Min	Max	
F_{REFCLK}	Refclk frequency	99.97	100.03	MHz
$T_{REFCLK-RMS-SRIS}$	RMS Refclk jitter for separate Refclk independent SSC architecture[1]		0.5	ps RMS
F_{SSC}	SSC frequency range	30	33	kHz
$T_{SSC-FREQ-DEVIATION}$	SSC deviation		+0.0/-0.5	%

Notes:

1. The Refclk jitter is measured after applying PLL transfer function (4 MHz, 2 dB) and CDR jitter transfer function defined in Figure 4-99.
2. The maximum rate of change of the clock frequency is 1250 ppm/µs.
3. The maximum rate of change of the clock phase is 3 ns/µs.

5

5. Power Management

This chapter describes PCI Express power management (PCI Express-PM) capabilities and protocols.

5.1. Overview

PCI Express-PM provides the following services:

- A mechanism to identify power management capabilities of a given Function
- The ability to transition a Function into a certain power management state
- Notification of the current power management state of a Function
- The option to wakeup the system on a specific event

PCI Express-PM is compatible with the *PCI Bus Power Management Interface Specification*, and the *Advanced Configuration and Power Interface Specification*. This chapter also defines PCI Express native power management extensions. These provide additional power management capabilities beyond the scope of the *PCI Bus Power Management Interface Specification*.

PCI Express-PM defines Link power management states that a PCI Express physical Link is permitted to enter in response to either software driven D-state transitions or active state Link power management activities. PCI Express Link states are not visible directly to legacy bus driver software, but are derived from the power management state of the components residing on those Links. Defined Link states are L0, L0s, L1, L2, and L3. The power savings increase as the Link state transitions from L0 through L3.

Components may wakeup the system using a wakeup mechanism followed by a power management event (PME) Message. PCI Express systems may provide the optional auxiliary power supply (Vaux) needed for wakeup operation from states where the main power supplies are off. PCI Express-PM extends the PME mechanism defined in conventional PCI-PM as PCI Express PME Messages include the Requester ID of the requesting agent. These PME Messages are in-band TLPs routed from the requesting Function towards the Root Complex.

Another distinction of the PCI Express-PM PME mechanism is its separation of the following two PME tasks:

- Reactivation (wakeup) of the associated resources (i.e., re-establishing reference clocks and main power rails to the PCI Express components)
- Sending a PME Message to the Root Complex

Active State Power Management (ASPM) is an autonomous hardware-based, active state mechanism that enables power savings even when the connected components are in the D0 state. After a period of idle Link time, an ASPM Physical-Layer protocol places the idle Link into a lower power state. Once in the lower-power state, transitions to the fully operative L0 state are triggered by traffic appearing on either side of the Link. ASPM may be disabled by software. Refer to Section 5.4.1 for more information on ASPM.

5.1.1. Statement of Requirements

All PCI Express Functions, with the exception of Functions in a Root Complex, are required to meet or exceed the minimum requirements defined by the PCI-PM software compatible PCI Express-PM features. Root Complexes are required to participate in Link power management DLLP protocols initiated by a Downstream device. For further details, refer to Section 5.3.2.

ASPM requirements are form factor specific. Refer to Section 5.4.1 for more information on ASPM.

5.2. Link State Power Management

PCI Express defines Link power management states, replacing the bus power management states that were defined by the *PCI Bus Power Management Interface Specification*. Link states are not visible to PCI-PM legacy compatible software, and are either derived from the power management D-states of the corresponding components connected to that Link or by ASPM protocols (see Section 5.4.1).

Note that the PCI Express Physical Layer may define additional intermediate states. Refer to Chapter 4 for more detail on each state and how the Physical Layer handles transitions between states.

PCI Express-PM defines the following Link power management states:

- L0 – Active state.

 L0 support is required for both ASPM and PCI-PM compatible power management.

 All PCI Express transactions and other operations are enabled.

- L0s – A low resume latency, energy saving "standby" state.

 L0s support is optional for ASPM unless the applicable form factor specification for the Link explicitly requires L0s support. It is not applicable to M-PCIe implementations. It is not applicable to PCI-PM compatible power management.

 All main power supplies, component reference clocks, and components' internal PLLs must be active at all times during L0s. TLP and DLLP transmission is disabled for a Port whose Link is in Tx_L0s.

 The Physical Layer provides mechanisms for quick transitions from this state to the L0 state. When common (distributed) reference clocks are used on both sides of a Link, the transition time from L0s to L0 is typically less than 100 Symbol Times.

 It is possible for the Transmit side of one component on a Link to be in L0s while the Transmit side of the other component on the Link is in L0.

- L1 – Higher latency, lower power "standby" state.

 L1 PM Substates functionality is not applicable and must not be enabled for M-PCIe Ports.

 L1 support is required for PCI-PM compatible power management. L1 is optional for ASPM unless specifically required by a particular form factor.

 When L1 PM Substates is enabled by setting one or more of the enable bits in the L1 PM Substates Control 1 Register this state is referred to as the L1.0 substate.

 All main power supplies must remain active during L1. As long as they adhere to the advertised L1 exit latencies, implementations are explicitly permitted to reduce power by applying techniques such as, but not limited to, periodic rather than continuous checking for Electrical Idle exit, checking for Electrical Idle exit on only one Lane, and powering off of unneeded circuits. All platform-provided component reference clocks must remain active during L1, except as permitted by Clock Power Management (using CLKREQ#) and/or L1 PM Substates when enabled. A component's internal PLLs may be shut off during L1, enabling greater power savings at a cost of increased exit latency.[61]

 The L1 state is entered whenever all Functions of a Downstream component on a given Link are programmed to a D-state other than D0. The L1 state also is entered if the Downstream component requests L1 entry (ASPM) and receives positive acknowledgement for the request.

 Exit from L1 is initiated by an Upstream-initiated transaction targeting a Downstream component, or by the Downstream component's initiation of a transaction heading Upstream. Transition from L1 to L0 is typically a few microseconds.

 TLP and DLLP transmission is disabled for a Link in L1.

Comment [VS164]: ECN: M-PCIe

Comment [VS165]: ECN: M-PCIe

Comment [VS166]: ECN: L1 PM Substates with CLKREQ

Comment [VS167]: ECN: L1 PM Substates with CLKREQ

[61] For example, disabling the internal PLL may be something that is desirable when in $D3_{hot}$, but not so when in D1 or D2.

L1 PM Substates – optional L1.1 and L1.2 substates of the L1 low power Link state for PCI-PM and ASPM.

In the L1.1 substate, the Link common mode voltages are maintained. The L1.1 substate is entered when the Link is in the L1.0 substate and conditions for entry into L1.1 substate are met. See Section 5.5.1 for details.

In the L1.2 substate, the Link common mode voltages are not required to be maintained. The L1.2 substate is entered when the Link is in the L1.0 substate and conditions for entry into L1.2 substate are met. See Section 5.5.1. for details.

Exit from all L1 PM Substates is initiated when the CLKREQ# signal is asserted (see Sections 5.5.2.1 and 5.5.3.3).

Comment [VS168]: ECN: L1 PM Substates with CLKREQ

- L2/L3 Ready – Staging point for L2 or L3.

 L2/L3 Ready transition protocol support is required.

 L2/L3 Ready is a pseudo-state (corresponding to the LTSSM L2 state) that a given Link enters when preparing for the removal of power and clocks from the Downstream component or from both attached components. This process is initiated after PM software transitions a device into a D3 state, and subsequently calls power management software to initiate the removal of power and clocks. After the Link enters the L2/L3 Ready state the component(s) are ready for power removal. After main power has been removed, the Link will either transition to L2 if Vaux is provided and used, or it will transition to L3 if no Vaux is provided or used. Note that these are PM pseudo-states for the Link; under these conditions, the LTSSM will in, general, operate only on main power, and so will power off with main power removal.

 The L2/L3 Ready state entry transition process must begin as soon as possible following the acknowledgment of a PME_Turn_Off Message, (i.e., the injection of a PME_TO_Ack TLP). The Downstream component initiates L2/L3 Ready entry by sending a PM_Enter_L23 DLLP. Refer to Section 5.7 for further detail on power management system Messages.

 TLP and DLLP transmission is disabled for a Link in L2/L3 Ready.

 Note: Exit from L2/L3 Ready back to L0 will be through intermediate LTSSM states. Refer to Chapter 4 for detailed information.

- L2 – Auxiliary-powered Link, deep-energy-saving state.

 L2 support is optional, and dependent upon the presence of Vaux.

 A component may only consume Vaux power if enabled to do so as described in Section 5.6.1.

 In L2, the component's main power supply inputs and reference clock inputs are shut off.

 When in L2, any Link reactivation wakeup logic (Beacon or WAKE#), PME context, and any other "keep alive" logic is powered by Vaux.

 TLP and DLLP transmission is disabled for a Link in L2.

- L3 – Link Off state.

 When no power is present, the component is in the L3 state.

- LDn – A transitional Link Down pseudo-state prior to L0.

This pseudo-state is associated with the LTSSM states Detect, Polling, and Configuration, and, when applicable, Disabled, Loopback, and Hot Reset.

Refer to Section 4.2 for further detail relating to entering and exiting each of the L-states between L0 and L2/L3 Ready (L2.Idle from the Chapter 4 perspective). The L2 state is an abstraction for PM purposes distinguished by the presence of auxiliary power, and should not be construed to imply a requirement that the LTSSM remain active.

The electrical section specifies the electrical properties of drivers and Receivers when no power is applied. This is the L3 state but the electrical section does not refer to L3.

Figure 5-1 highlights the legitimate L-state transitions that may occur during the course of Link operation.

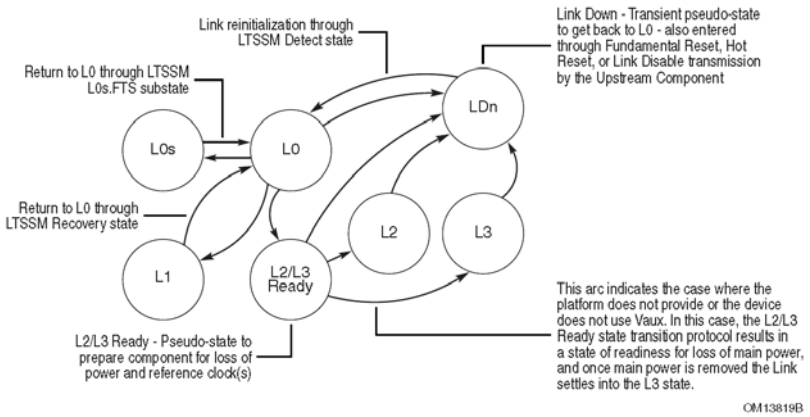


Figure 5-1: Link Power Management State Flow Diagram

The L1 and L2/L3 Ready entry negotiations happen while in the L0 state. L1 and L2/L3 Ready are entered only after the negotiation completes. Link Power Management remains in L0 until the negotiation process is completed, unless LDn occurs. Note that these states and state transitions do not correspond directly to the actions of the Physical Layer LTSSM. For example in Figure 5-1, L0 encompasses the LTSSM L0, Recovery, and, during LinkUp, Configuration states. Also, the LTSSM is typically powered by main power (not Vaux), so LTSSM will not be powered in either the L2 or the L3 state.

The following example sequence illustrates the multi-step Link state transition process leading up to entering a system sleep state:

1. System software directs all Functions of a Downstream component to $D3_{hot}$.
2. The Downstream component then initiates the transition of the Link to L1 as required.
3. System software then causes the Root Complex to broadcast the PME_Turn_Off Message in preparation for removing the main power source.
4. This Message causes the subject Link to transition back to L0 in order to send it and to enable the Downstream component to respond with PME_TO_Ack.

5. After sending the PME_TO_Ack, the Downstream component initiates the L2/L3 Ready transition protocol.

L0 → L1 → L0 → L2/L3 Ready

As the following example illustrates, it is also possible to remove power without first placing all Functions into $D3_{hot}$:

1. System software causes the Root Complex to broadcast the PME_Turn_Off Message in preparation for removing the main power source.
2. The Downstream components respond with PME_TO_Ack.
3. After sending the PME_TO_Ack, the Downstream component initiates the L2/L3 Ready transition protocol.

L0 → L2/L3 Ready

The L1 entry negotiation (whether invoked via PCI-PM or ASPM mechanisms) and the L2/L3 Ready entry negotiation map to a state machine which corresponds to the actions described later in this chapter. This state machine is reset to an idle state. For a Downstream component, the first action taken by the state machine, after leaving the idle state, is to start sending the appropriate entry DLLPs depending on the type of negotiation. If the negotiation is interrupted, for example by a trip through Recovery, the state machine in both components is reset back to the idle state. ~~For t~~The Upstream component~~, this~~ must always ~~means to~~ go to the idle state, and wait to receive entry DLLPs. ~~For t~~The Downstream component~~, this means~~must always go to the idle state and must always proceed to sending entry DLLPs to restart the negotiation.

Comment [VS169]: Errata A16

Table 5-1 summarizes each L-state, describing when they are used, and the platform and component behaviors that correspond to each.

A "Yes" entry indicates that support is required (unless otherwise noted). "On" and "Off" entries indicate the required clocking and power delivery. "On/Off" indicates an optional design choice.

Table 5-1: Summary of PCI Express Link Power Management States

	L-State Description	Used by S/W Directed PM	Used by ASPM	Platform Reference Clocks	Platform Main Power	Component Internal PLL	Platform Vaux
L0	Fully active Link	Yes (D0)	Yes (D0)	On	On	On	On/Off
L0s	Standby state	No	Yes [1] (opt., D0)	On	On	On	On/Off
L1	Lower power standby	Yes ($D1-D3_{hot}$)	Yes (opt., D0)	On/Off [6]	On	On/Off [2]	On/Off
L2/L3 Ready (pseudo -state)	Staging point for power removal	Yes [3]	No	On/Off [6]	On	On/Off	On/Off
L2	Low power sleep state (all clocks, main power off)	Yes [4]	No	Off	Off	Off	On [5]
L3	Off (zero power)	n/a	n/a	Off	Off	Off	Off
LDn (pseudo -state)	Transitional state preceding L0	Yes	N/A	On	On	On/Off	On/Off

Notes:
1. L0s exit latency will be greatest in Link configurations with independent reference clock inputs for components connected to opposite ends of a given Link (vs. a common, distributed reference clock).
2. L1 exit latency will be greatest for components that internally shut off their PLLs during this state.
3. L2/L3 Ready entry sequence is initiated at the completion of the PME_Turn_Off/PME_TO_Ack protocol handshake. It is not directly affiliated with either a D-State transition or a transition in accordance with ASPM policies and procedures.
4. Depending upon the platform implementation, the system's sleep state may use the L2 state, transition to fully off (L3), or it may leave Links in the L2/L3 Ready state. L2/L3 Ready state transition protocol is initiated by the Downstream component following reception and TLP acknowledgement of the PME_Turn_Off TLP Message. While platform support for an L2 sleep state configuration is optional (depending on the availability of Vaux), component protocol support for transitioning the Link to the L2/L3 Ready state is required.
5. L2 is distinguished from the L3 state only by the presence and use of Vaux. After the completion of the L2/L3 Ready state transition protocol and before main power has been removed, the Link has indicated its readiness for main power removal.
6. Low-power mobile or handheld devices may reduce power by clock gating the reference clock(s) via the "clock request" (CLKREQ#) mechanism. As a result, components targeting these devices should be tolerant of the additional delays required to re-energize the reference clock during the low-power state exit.

5.3. PCI-PM Software Compatible Mechanisms

5.3.1. Device Power Management States (D-States) of a Function

PCI Express supports all PCI-PM device power management states. All Functions must support the D0 and D3 states (both $D3_{hot}$ and $D3_{cold}$). The D1 and D2 states are optional. Refer to the *PCI Bus Power Management Interface Specification* for further detail relating to the PCI-PM compatible features described in this specification. Note that where this specification defines detail that departs from the *PCI Bus Power Management Interface Specification*, this specification takes precedence for components and Link hierarchies.



IMPLEMENTATION NOTE

Switch and Root Port Virtual Bridge Behavior in Non-D0 States

When a Type 1 Function associated with a Switch/Root Port (a "virtual bridge") is in a non-D0 power state, it will emulate the behavior of a conventional PCI bridge in its handling of Memory, I/O, and Configuration Requests and Completions. All Memory and I/O requests flowing Downstream are terminated as Unsupported Requests. All Type 1 Configuration Requests are terminated as Unsupported Requests, however Type 0 Configuration Request handling is unaffected by the virtual bridge D state. Completions flowing in either direction across the virtual bridge are unaffected by the virtual bridge D state.

Note that the handling of Messages is not affected by the PM state of the virtual bridge.

5.3.1.1. D0 State

All Functions must support the D0 state. D0 is divided into two distinct substates, the "un-initialized" substate and the "active" substate. When a component comes out of Conventional Reset ~~or FLR, it defaults to~~ all Functions of the component enter the $D0_{uninitialized}$ state. When a Function completes FLR, it enters the $D0_{uninitialized}$ state. ~~Components that are in this state will be enumerated and configured by the Hierarchy enumeration process. Following the completion of the enumeration and~~ After configuration is complete a ~~process the~~ Function enters the $D0_{active}$ state, the fully operational state for a PCI Express Function. A Function enters the $D0_{active}$ state whenever any single or combination of the Function's Memory Space Enable, I/O Space Enable, or Bus Master Enable bits have been Set~~enabled by system software~~[62].

Comment [VS170]: Errata A14

5.3.1.2. D1 State

D1 support is optional. While in the D1 state, a Function must not initiate any Request TLPs on the Link with the exception of a PME Message as defined in Section 5.3.3. Configuration and Message Requests are the only TLPs accepted by a Function in the D1 state. All other received Requests must be handled as Unsupported Requests, and all received Completions may optionally

[62] A Function remains in $D0_{active}$ even if these enable bits are subsequently cleared.

be handled as Unexpected Completions. If an error caused by a received TLP (e.g., an Unsupported Request) is detected while in D1, and reporting is enabled, the Link must be returned to L0 if it is not already in L0 and an error Message must be sent. If an error caused by an event other than a received TLP (e.g., a Completion Timeout) is detected while in D1, an error Message must be sent when the Function is programmed back to the D0 state.

Note that a Function's software driver participates in the process of transitioning the Function from D0 to D1. It contributes to the process by saving any functional state (if necessary), and otherwise preparing the Function for the transition to D1. As part of this quiescence process the Function's software driver must ensure that any mid-transaction TLPs (i.e., Requests with outstanding Completions), are terminated prior to handing control to the system configuration software that would then complete the transition to D1.

5.3.1.3. D2 State

D2 support is optional. While in the D2 state, a Function must not initiate any Request TLPs on the Link with the exception of a PME Message as defined in Section 5.3.3. Configuration and Message requests are the only TLPs accepted by a Function in the D2 state. All other received Requests must be handled as Unsupported Requests, and all received Completions may optionally be handled as Unexpected Completions. If an error caused by a received TLP (e.g., an Unsupported Request) is detected while in D2, and reporting is enabled, the Link must be returned to L0 if it is not already in L0 and an error Message must be sent. If an error caused by an event other than a received TLP (e.g., a Completion Timeout) is detected while in D2, an error Message must be sent when the Function is programmed back to the D0 state.

Note that a Function's software driver participates in the process of transitioning the Function from D0 to D2. It contributes to the process by saving any functional state (if necessary), and otherwise preparing the Function for the transition to D2. As part of this quiescence process the Function's software driver must ensure that any mid-transaction TLPs (i.e., Requests with outstanding Completions), are terminated prior to handing control to the system configuration software that would then complete the transition to D2.

5.3.1.4. D3 State

D3 support is required, (both the $D3_{cold}$ and the $D3_{hot}$ states). Functions supporting PME generation from D3 must support it for both $D3_{cold}$ and the $D3_{hot}$ states.

Functional context is required to be maintained by Functions in the $D3_{hot}$ state if the No_Soft_Reset field in the PMCSR is Set. In this case, software is not required to re-initialize the Function after a transition from $D3_{hot}$ to D0 (the Function will be in the $D0_{active}$ state). If the No_Soft_Reset bit is Clear, functional context is not required to be maintained by the Function in the $D3_{hot}$ state. As a result, in this case software is required to fully re-initialize the Function after a transition to D0 as the Function will be in the $D0_{uninitialized}$ state.

The Function will be reset if the Link state has transitioned to the L2/L3 Ready state regardless of the value of the No_Soft_Reset bit.

IMPLEMENTATION NOTE

Transitioning to L2/L3 Ready

As described in Section 5.2, transition to the L2/L3 Ready state is initiated by platform power management software in order to begin the process of removing main power and clocks from the device. As a result, it is expected that a device will transition to $D3_{cold}$ shortly after its Link transitions to L2/L3 Ready, making the No_Soft_Reset bit, which only applies to $D3_{hot}$, irrelevant. While there is no guarantee of this correlation between L2/L3 Ready and $D3_{cold}$, system software should ensure that the L2/L3 Ready state is entered only when the intent is to remove device main power. Device Functions, including those that are otherwise capable of maintaining functional context while in $D3_{hot}$ (i.e., set the No_Soft_Reset bit), are required to re-initialize internal state as described in Section 2.9.1 when exiting L2/L3 Ready due to the required DL_Down status indication.

Unless Readiness Notifications mechanisms are used (See Section 6.23) ~~Unless the Immediate_Readiness_on_Return_to_D0 bit in the PCI-PM Power Management Capabilities Register is Set~~, sSystem software must allow a minimum recovery time following a $D3_{hot} \rightarrow D0$ transition of at least 10 ms, prior to accessing the Function. This recovery time may, for example, be used by the $D3_{hot} \rightarrow D0$ transitioning component to bootstrap any of its component interfaces (e.g., from serial ROM) prior to being accessible. Attempts to target the Function during the recovery time (including configuration request packets) will result in undefined behavior.

Comment [VS171]: Errata A27

Comment [VS172]: D-ECN-Readiness Notifications

5.3.1.4.1. $D3_{hot}$ State

Configuration and Message requests are the only TLPs accepted by a Function in the $D3_{hot}$ state. All other received Requests must be handled as Unsupported Requests, and all received Completions may optionally be handled as Unexpected Completions. If an error caused by a received TLP (e.g., an Unsupported Request) is detected while in $D3_{hot}$, and reporting is enabled, the Link must be returned to L0 if it is not already in L0 and an error Message must be sent. If an error caused by an event other than a received TLP (e.g., a Completion Timeout) is detected while in $D3_{hot}$, an error Message may optionally be sent when the Function is programmed back to the D0 state. Once in $D3_{hot}$ the Function can later be transitioned into $D3_{cold}$ (by removing power from its host component).

Note that a Function's software driver participates in the process of transitioning the Function from D0 to $D3_{hot}$. It contributes to the process by saving any functional state that would otherwise be lost with removal of main power, and otherwise preparing the Function for the transition to $D3_{hot}$. As part of this quiescence process the Function's software driver must ensure that any outstanding transactions (i.e., Requests with outstanding Completions), are terminated prior to handing control to the system configuration software that would then complete the transition to $D3_{hot}$.

Note that $D3_{hot}$ is also a useful state for reducing power consumption by idle components in an otherwise running system.

Functions that are in $D3_{hot}$ may be transitioned by software (writing to their PMCSR PowerState field) to the $D0_{active}$ state or the $D0_{uninitialized}$ state. Note that the Function is not required to generate an internal hardware reset during or immediately following its transition from $D3_{hot}$ to D0 (see usage of the No_Soft_Reset bit in the PMCSR).



IMPLEMENTATION NOTE

Multi-Function Device Issues with Soft Reset

With multi-Function devices (MFDs), certain control settings affecting overall device behavior are determined either by the collective settings in all Functions or strictly off the settings in Function 0. Here are some key examples:

- With non-ARI MFDs, certain controls in the Device Control register and Link Control registers operate off the collective settings of all Functions (see Section 7.8.4 and Section 7.8.7).
- With ARI Devices, certain controls in the Device Control register and Link Control registers operate strictly off the settings in Function 0 (see Section 7.8.4 and Section 7.8.7).
- With all MFDs, certain controls in the Device Control 2 and Link Control 2 registers operate strictly off the settings in Function 0 (see Section 7.8.16 and Section 7.8.19).

Performing a soft reset on any Function (especially Function 0) may disrupt the proper operation of other active Functions in the MFD. Since some Operating Systems transition a given Function between $D3_{hot}$ and D0 with the expectation that other Functions will not be impacted, it is strongly recommended that every Function in an MFD be implemented with the No_Soft_Reset bit Set in the Power Management Status/Control register. This way, transitioning a given Function from $D3_{hot}$ to D0 will not disrupt the proper operation of other active Functions.

It is also strongly recommended that every Endpoint Function in an MFD implement Function Level Reset (FLR). FLR can be used to reset an individual Endpoint Function without impacting the settings that might affect other Functions, particularly if those Functions are active. As a result of FLR's quiescing, error recovery, and cleansing for reuse properties, FLR is also recommended for single-Function Endpoint devices.

5.3.1.4.2. $D3_{cold}$ State

A Function transitions to the $D3_{cold}$ state when its main power is removed. A power-on sequence with its associated cold reset transitions a Function from the $D3_{cold}$ state to the $D0_{uninitialized}$ state. At this point, software must perform a full initialization of the Function in order to re-establish all functional context, completing the restoration of the Function to its $D0_{active}$ state.

Functions that support wakeup functionality from $D3_{cold}$ must maintain their PME context (in the PMCSR) for inspection by PME service routine software during the course of the resume process.



IMPLEMENTATION NOTE

PME Context

Examples of PME context include, but are not limited to, a Function's PME_Status bit, the requesting agent's Requester ID, Caller ID if supported by a modem, IP information for IP directed network packets that trigger a resume event, etc.

A Function's PME assertion is acknowledged when system software performs a "write 1 to clear" configuration transaction to the asserting Function's PME_Status bit of its PCI-PM compatible PMCSR.

An auxiliary power source must be used to support PME event detection within a Function, Link reactivation, and to preserve PME context from within $D3_{cold}$. Note that once the I/O Hierarchy has been brought back to a fully communicating state, as a result of the Link reactivation, the waking agent then propagates a PME Message to the root of the Hierarchy indicating the source of the PME event. Refer to Section 5.3.3 for further PME specific detail.

5.3.2. PM Software Control of the Link Power Management State

The power management state of a Link is determined by the D-state of its Downstream component.

Table 5-2 depicts the relationships between the power state of a component (with an Upstream Port) and its Upstream Link.

Table 5-2: Relation Between Power Management States of Link and Components

Downstream Component D-State	Permissible Upstream Component D-State	Permissible Interconnect State
D0	D0	L0, L0s, L1[(1)], L2/L3 Ready
D1	D0-D1	L1, L2/L3 Ready
D2	D0-D2	L1, L2/L3 Ready
$D3_{hot}$	D0-$D3_{hot}$	L1, L2/L3 Ready
$D3_{cold}$	D0-$D3_{cold}$	L2[(2)], L3

Notes:
1. Requirements for ASPM L0s and ASPM L1 support are form factor specific.
2. If Vaux is provided by the platform, the Link sleeps in L2. In the absence of Vaux, the L-state is L3.

The following rules relate to PCI-PM compatible power management:

- Devices in D0, D1, D2, and $D3_{hot}$ must respond to the receipt of a PME_Turn_Off Message by the transmission of a PME_TO_Ack Message.
- In any device D state, following the execution of a PME_Turn_Off/PME_TO_Ack handshake sequence, a Downstream component must request a Link transition to L2/L3 Ready using the PM_Enter_L23 DLLP. Following the L2/L3 Ready entry transition protocol the Downstream component must be ready for loss of main power and reference clock.
- The Upstream Port of a single-Function device must initiate a Link state transition to L1 based solely upon its Function being programmed to D1, D2, or $D3_{hot}$. In the case of the Switch, system software bears the responsibility of ensuring that any D-state programming of a Switch's Upstream Port is done in a compliant manner with respect to hierarchy-wide PM policies (i.e., the Upstream Port cannot be programmed to a D-state that is any less active than the most active Downstream Port and Downstream connected component/Function(s)).

- The Upstream Port of a non-ARI multi-Function device must not initiate a Link state transition to L1 (on behalf of PCI-PM) until all of its Functions have been programmed to a non-D0 D-state.
- The Upstream Port of an ARI Device must not initiate a Link state transition to L1 (on behalf of PCI-PM) until at least one of its Functions has been programmed to a non-D0 state, and all of its Functions are either in a non-D0 state or the $D0_{uninitialized}$ state.

5.3.2.1. Entry into the L1 State

Figure 5-2 depicts the process by which a Link transitions into the L1 state as a direct result of power management software programming the Downstream connected component into a lower power state, (either D1, D2, or $D3_{hot}$ state). This figure and the subsequent description outline the transition process for a single -Function Downstream component that is being programmed to a non-D0 state.

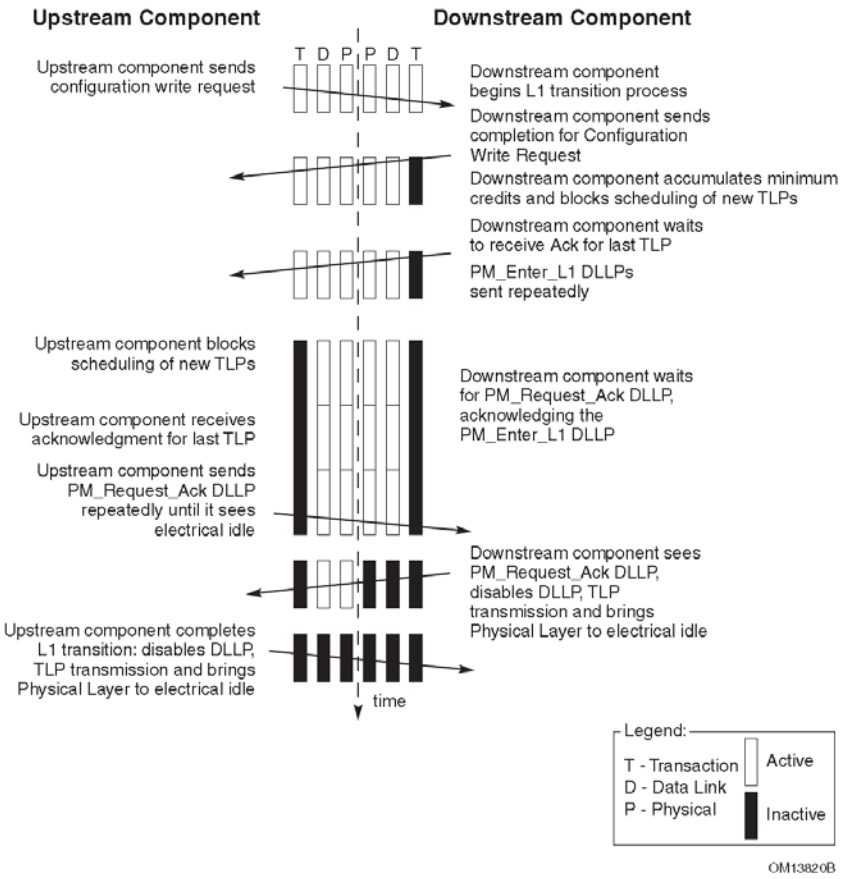


Figure 5-2: Entry into the L1 Link State

The following text provides additional detail for the Link state transition process shown in Figure 5-2.

PM Software Request:

1. PM software sends a Configuration Write Request TLP to the Downstream Function's PMCSR to change the Downstream Function's D-state (from D0 to D1 for example).

Downstream Component Link State Transition Initiation Process:

2. The Downstream component schedules the Completion corresponding to the Configuration Write Request to its PMCSR PowerState field and accounts for the completion credits required.
3. The Downstream component must then wait until it accumulates at least the minimum number of credits required to send the largest possible packet for any FC type (if it does not already have such credits). All Transaction Layer TLP scheduling is then suspended.
4. The Downstream component then waits until it receives a Link Layer acknowledgement for the PMCSR Write Completion, and any other TLPs it had previously sent. The component must retransmit a TLP out of its Data Link Layer Retry buffer if required to do so by Data Link Layer rules.
5. Once all of the Downstream components' TLPs have been acknowledged, the Downstream component starts to transmit PM_Enter_L1 DLLPs. The Downstream component sends this DLLP repeatedly with no more than four (when using 8b/10b encoding) or 16 (when using 128b/130b encoding) Symbol times of idle between subsequent transmissions of the PM_Enter_L1 DLLP. The transmission of other DLLPs and SKP Ordered Sets is permitted at any time between PM_Enter_L1 transmissions, and do not contribute to this idle time limit.

 The Downstream component continues to transmit the PM_Enter_L1 DLLP as described above until it receives a response from the Upstream component[63] (PM_Request_Ack).

 The Downstream component must continue to accept TLPs and DLLPs from the Upstream component, and continue to respond with DLLPs, including FC update DLLPs and Ack/Nak DLLPs, as required. Any TLPs that are blocked from transmission (including responses to TLP(s) received) must be stored for later transmission, and must cause the Downstream component to initiate L1 exit as soon as possible following L1 entry.

Upstream Component Link State Transition Process:

6. Upon receiving the PM_Enter_L1 DLLP, the Upstream component blocks the scheduling of all TLP transmissions.
7. The Upstream component then must wait until it receives a Link Layer acknowledgement for the last TLP it had previously sent. The Upstream component must retransmit a TLP from its Link Layer retry buffer if required to do so by the Link Layer rules.
8. Once all of the Upstream component's TLPs have been acknowledged, the Upstream component must send PM_Request_Ack DLLPs Downstream, regardless of any outstanding Requests. The Upstream component sends this DLLP repeatedly with no more than four (when

[63] If at this point the Downstream component needs to initiate a transfer on the Link, it must first complete the transition to L1. Once in L1 it is then permitted to initiate an exit L1 to handle the transfer.

using 8b/10b encoding) or 16 (when using 128b/130b encoding) Symbol times of idle between subsequent transmissions of the PM_Request_Ack DLLP. The transmission of SKP Ordered Sets is permitted at any time between PM_Request_Ack transmissions, and does not contribute to this idle time limit.

The Upstream component continues to transmit the PM_Request_Ack DLLP as described above until it observes its receive Lanes enter into the Electrical Idle state. Refer to Chapter 4 for more details on the Physical Layer behavior.

Completing the L1 Link State Transition:

9. Once the Downstream component has captured the PM_Request_Ack DLLP on its Receive Lanes (signaling that the Upstream component acknowledged the transition to L1 request), it then disables DLLP transmission and brings the Upstream directed physical Link into the Electrical Idle state.
10. When the Receive Lanes on the Upstream component enter the Electrical Idle state, the Upstream component stops sending PM_Request_Ack DLLPs, disables DLLP transmission, and brings its Transmit Lanes to Electrical Idle completing the transition of the Link to L1.

 When two components' interconnecting Link is in L1 as a result of the Downstream component being programmed to a non-D0 state, both components suspend the operation of their Flow Control Update and, if implemented, Update FCP Timer (see Section 2.6.1.2) counter mechanisms. Refer to Chapter 4 for more detail on the Physical Layer behavior.

Refer to Section 5.2 if the negotiation to L1 is interrupted.

Components on either end of a Link in L1 may optionally disable their internal PLLs in order to conserve more energy. Note, however, that platform supplied main power and reference clocks must continue to be supplied to components on both ends of an L1 Link in the L1.0 substate of L1.

Refer to Section 5.6.1 for entry into the L1 PM Substates.

Comment [VS173]: ECN: L1 PM Substates with CLKREQ

5.3.2.2. Exit from L1 State

L1 exit can be initiated by the component on either end of a Link.

Upon exit from L1, it is recommended that the Downstream component send flow control update DLLPs for all enabled VCs and FC types starting within 1 µs of L1 exit.

The physical mechanism for transitioning a Link from L1 to L0 is described in detail in Chapter 4.

L1 exit must be initiated by a component if that component needs to transmit a TLP on the Link. An Upstream component must initiate L1 exit on a Downstream Port even if it does not have the flow control credits needed to transmit the TLP that it needs to transmit. Following L1 exit, the Upstream component must wait to receive the needed credit from the Downstream component. Figure 5-3 outlines an example sequence that would trigger an Upstream component to initiate transition of the Link to the L0 state.

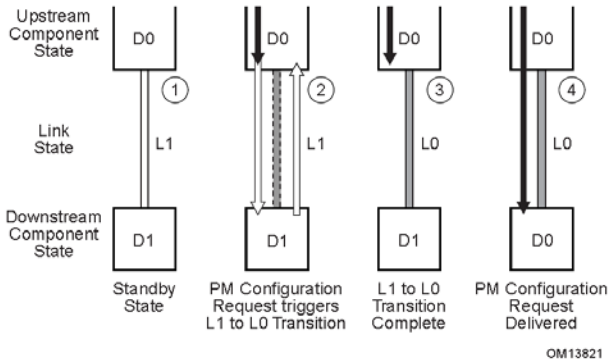


Figure 5-3: Exit from L1 Link State Initiated by Upstream Component

Sequence of events:

1. Power management software initiates a configuration cycle targeting a PM configuration register (the PowerState field of the PMCSR in this example) within a Function that resides in the Downstream component (e.g., to bring the Function back to the D0 state).
2. The Upstream component detects that a configuration cycle is intended for a Link that is currently in a low power state, and as a result, initiates a transition of that Link into the L0 state.
3. If the Link is in either L1.1 or L1.2 substates of L1, then the Upstream component initiates a transition of the Link into the L1.0 substate of L1.
4. In accordance with the Chapter 4 definition, both directions of the Link enter into Link training, resulting in the transition of the Link to the L0 state. The L1 → L0 transition is discussed in detail in Chapter 4.
5. Once both directions of the Link are back to the active L0 state, the Upstream Port sends the configuration Packet Downstream.

Comment [VS174]: ECN: L1 PM Substates with CLKREQ

5.3.2.3. Entry into the L2/L3 Ready State

Transition to the L2/L3 Ready state follows a process that is similar to the L1 entry process. There are some minor differences between the two that are spelled out below.

- L2/L3 Ready entry transition protocol does not immediately result in an L2 or L3 Link state. The transition to L2/L3 Ready is effectively a handshake to establish the Downstream component's readiness for power removal. L2 or L3 is ultimately achieved when the platform removes the components' power and reference clock.
- The time for L2/L3 Ready entry transition is indicated by the completion of the PME_Turn_Off/PME_TO_Ack handshake sequence. Any actions on the part of the Downstream component necessary to ready itself for loss of power must be completed prior to initiating the transition to L2/L3 Ready. Once all preparations for loss of power and clock are completed, L2/L3 Ready entry is initiated by the Downstream component by sending the PM_Enter_L23 DLLP Upstream.

- L2/L3 Ready entry transition protocol uses the PM_Enter_L23 DLLP.

 Note that the PM_Enter_L23 DLLPs are sent continuously until an acknowledgement is received or power is removed.

- Refer to Section 5.2 if the negotiation to L2/L3 Ready is interrupted.

5.3.3. Power Management Event Mechanisms

5.3.3.1. Motivation

The PCI Express PME mechanism is software compatible with the PME mechanism defined by the *PCI Bus Power Management Interface Specification*. Power Management Events are generated by Functions as a means of requesting a PM state change. Power Management Events are typically utilized to revive the system or an individual Function from a low power state.

Power management software may transition a Hierarchy into a low power state, and transition the Upstream Links of these devices into the non-communicating L2 state.[64] The PCI Express PME generation mechanism is, therefore, broken into two components:

- Waking a non-communicating Hierarchy (wakeup). This step is required only if the Upstream Link of the device originating the PME is in the non-communicating L2 state, since in that state the device cannot send a PM_PME Message Upstream.
- Sending a PM_PME Message to the root of the Hierarchy

PME indications that originate from PCI Express Endpoints or PCI Express Legacy Endpoints are propagated to the Root Complex in the form of TLP messages. PM_PME Messages identify the requesting agent within the Hierarchy (via the Requester ID of the PME Message header). Explicit identification within the PM_PME Message is intended to facilitate quicker PME service routine response, and hence shorter resume time.

If a Root Complex Event Collector is implemented, PME indications that originate from a Root Complex Integrated Endpoint may optionally be reported in a Root Complex Event Collector residing on the same Logical Bus as the Root Complex Integrated Endpoint. The Root Complex Event Collector must explicitly declare supported Root Complex Integrated Endpoints as part of its capabilities; each Root Complex Integrated Endpoint must be associated with no more than one Root Complex Event Collector. Root Complex Event Collectors explicitly identify the logical location of the requesting agent to facilitate quicker PME service routine response.

PME indications that originate from a Root Port itself are reported through the same Root Port.

5.3.3.2. Link Wakeup

The Link wakeup mechanisms provide a means of signaling the platform to re-establish power and reference clocks to the components within its domain. There are two defined wakeup mechanisms: Beacon and WAKE#. The Beacon mechanism uses in-band signaling to implement wakeup functionality, and is described in Section 4.3.5.7. For components that support wakeup

64 The L2 state is defined as "non-communicating" since component reference clock and main power supply are removed in that state.

functionality, the form factor specification(s) targeted by the implementation determine the support requirements for the wakeup mechanism. Switch components targeting applications where Beacon is used on some Ports of the Switch and WAKE# is used for other Ports must translate the wakeup mechanism appropriately (see the implementation note entitled "Example of WAKE# to Beacon Translation" on page 459). In applications where WAKE# is the only wakeup mechanism used, the Root Complex is not required to support the receipt of Beacon.

The WAKE# mechanism uses sideband signaling to implement wakeup functionality. WAKE# is an "open drain" signal asserted by components requesting wakeup and observed by the associated power controller. WAKE# is only defined for certain form factors, and the detailed specifications for WAKE# are included in the relevant form factor specifications. Specific form factor specifications may require the use of either Beacon or WAKE# as the wakeup mechanism.

When WAKE# is used as a wakeup mechanism, once WAKE# has been asserted, the asserting Function must continue to drive the signal low until main power has been restored to the component as indicated by Fundamental Reset going inactive.

The system is not required to route or buffer WAKE# in such a way that an Endpoint is guaranteed to be able to detect that the signal has been asserted by another Function.

Before using any wakeup mechanism, a Function must be enabled by software to do so by setting the Function's PME_En bit in the PMCSR. The PME_Status bit is sticky, and Functions must maintain the value of the PME_Status bit through reset if Aux power is available and they are enabled for wakeup events (this requirement also applies to the PME_En bit in the PMCSR and the Aux Power PM Enable bit in the Device Control register).

Systems that allow PME generation from $D3_{cold}$ state must provide auxiliary power to support Link wakeup when the main system power rails are off. A component may only consume auxiliary power if software has enabled it to do so as described in Section 5.6.1. Software is required to enable auxiliary power consumption in all components that participate in Link wakeup, including all components that must propagate the Beacon signal. In the presence of legacy system software, this is the responsibility of system firmware.

Regardless of the wakeup mechanism used, once the Link has been re-activated and trained, the requesting agent then propagates a PM_PME Message Upstream to the Root Complex. From a power management point of view, the two wakeup mechanisms provide the same functionality, and are not distinguished elsewhere in this chapter.

IMPLEMENTATION NOTE

Example of WAKE# to Beacon Translation

Switch components targeting applications that connect "Beacon domains" and "WAKE# domains" must translate the wakeup mechanism appropriately. Figure 5-4 shows two example systems, each including slots that use the WAKE# wakeup mechanism. In Case 1, WAKE# is input directly to the Power Management Controller, and no translation is required. In Case 2, WAKE# is an input to the Switch, and in response to WAKE# being asserted the Switch must generate a Beacon that is propagated to the Root Complex/Power Management Controller.

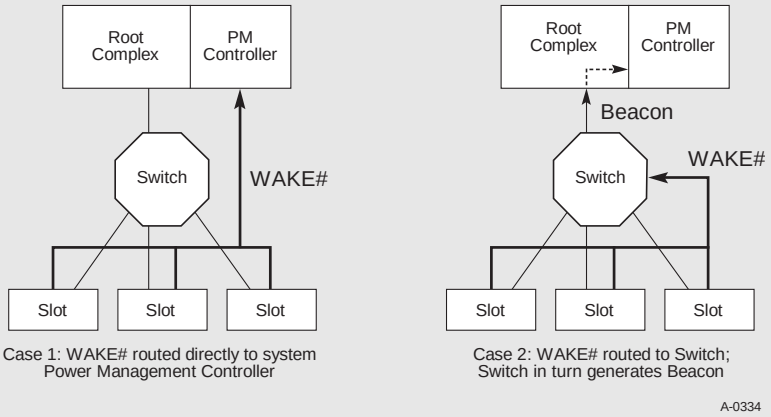


Figure 5-4: Conceptual Diagrams Showing Two Example Cases of WAKE# Routing

5.3.3.2.1. PME Synchronization

PCI Express-PM introduces a fence mechanism that serves to initiate the power removal sequence while also coordinating the behavior of the platform's power management controller and PME handling by PCI Express agents.

PME_Turn_Off Broadcast Message

Before main component power and reference clocks are turned off, the Root Complex or Switch Downstream Port must issue a broadcast Message that instructs all agents Downstream of that point within the hierarchy to cease initiation of any subsequent PM_PME Messages, effective immediately upon receipt of the PME_Turn_Off Message.

Each PCI Express agent is required to respond with a TLP "acknowledgement" Message, PME_TO_Ack that is always routed Upstream. In all cases, the PME_TO_Ack Message must terminate at the PME_Turn_Off Message's point of origin.[65]

[65] Point of origin for the PME_Turn_Off Message could be all of the Root Ports for a given Root Complex (full platform sleep state transition), an individual Root Port, or a Switch Downstream Port.

A Switch must report an "aggregate" acknowledgement only after having received PME_TO_Ack Messages from each of its Downstream Ports. Once a PME_TO_Ack Message has arrived on each Downstream Port, the Switch must then send a PME_TO_Ack packet on its Upstream Port. The occurrence of any one of the following must reset the aggregation mechanism: the transmission of the PME_TO_Ack Message from the Upstream Port, the receipt of any TLP at the Upstream Port, the removal of main power to the Switch, or Fundamental Reset.

All components with an Upstream Port must accept and acknowledge the PME_Turn_Off Message regardless of the D state of the associated device or any of its Functions for a multi-Function device. Once a component has sent a PME_TO_Ack Message, it must then prepare for removal of its power and reference clocks by initiating a transition to the L2/L3 Ready state.

A Switch must transition its Upstream Link to the L2/L3 Ready state after all of its Downstream Ports have entered the L2/L3 Ready state.

The Links attached to the originator of the PME_Turn_Off Message are the last to assume the L2/L3 Ready state. This state transition serves as an indication to the power delivery manager[66] that all Links within that portion of the Hierarchy have successfully retired all in flight PME Messages to the point of PME_Turn_Off Message origin and have performed any necessary local conditioning in preparation for power removal.

In order to avoid deadlock in the case where one or more devices do not respond with a PME_TO_Ack Message and then put their Links into the L2/L3 Ready state, the power manager must implement a timeout after waiting for a certain amount of time, after which it proceeds as if the Message had been received and all Links put into the L2/L3 Ready state. The recommended limit for this timer is in the range of 1 ms to 10 ms.

The power delivery manager must wait a minimum of 100 ns after observing all Links corresponding to the point of origin of the PME_Turn_Off Message enter L2/L3 Ready before removing the components' reference clock and main power. This requirement does not apply in the case where the above mentioned timer triggers.

[66] Power delivery control within this context relates to control over the entire Link hierarchy, or over a subset of Links ranging down to a single Link and associated Endpoint for sub hierarchies supporting independently managed power and clock distribution.



IMPLEMENTATION NOTE

PME_TO_Ack Message Proxy by Switches

One of the PME_Turn_Off/PME_TO_Ack handshake's key roles is to ensure that all in flight PME Messages are flushed from the PCI Express fabric prior to sleep state power removal. This is guaranteed to occur because PME Messages and the PME_TO_Ack Messages both use the posted request queue within VC0 and so all previously injected PME Messages will be made visible to the system before the PME_TO_Ack is received at the Root Complex. Once all Downstream Ports of the Root Complex receive a PME_TO_Ack Message the Root Complex can then signal the power manager that it is safe to remove power without loss of any PME Messages.

Switches create points of hierarchical expansion and, therefore, must wait for all of their connected Downstream Ports to receive a PME_TO_Ack Message before they can send a PME_TO_Ack Message Upstream on behalf of the sub-hierarchy that it has created Downstream. This can be accomplished very simply using common score boarding techniques. For example, once a PME_Turn_Off broadcast Message has been broadcast Downstream of the Switch, the Switch simply checks off each Downstream Port having received a PME_TO_Ack. Once the last of its active Downstream Ports receives a PME_TO_Ack, the Switch will then send a single PME_TO_Ack Message Upstream as a proxy on behalf of the entire sub-hierarchy Downstream of it. Note that once a Downstream Port receives a PME_TO_Ack Message and the Switch has scored its arrival, the Port is then free to drop the packet from its internal queues and free up the corresponding posted request queue FC credits.

5.3.3.3. PM_PME Messages

PM_PME Messages are posted Transaction Layer Packets (TLPs) that inform the power management software which agent within the Hierarchy requests a PM state change. PM_PME Messages, like all other Power Management system Messages, must use the general purpose Traffic Class, TC #0.

PM_PME Messages are always routed in the direction of the Root Complex. To send a PM_PME Message on its Upstream Link, a device must transition the Link to the L0 state (if the Link was not in that state already). Unless otherwise noted, the device will keep the Link in the L0 state following the transmission of a PM_PME Message.

5.3.3.3.1. PM_PME "Backpressure" Deadlock Avoidance

A Root Complex is typically implemented with local buffering to store temporarily a finite number of PM_PME Messages that could potentially be simultaneously propagating through the Hierarchy. Given a limited number of PM_PME Messages that can be stored within the Root Complex, there can be backpressure applied to the Upstream directed posted queue in the event that the capacity of this temporary PM_PME Message buffer is exceeded.

Deadlock can occur according to the following example scenario:

1. Incoming PM_PME Messages fill the Root Complex's temporary storage to its capacity while there are additional PM_PME Messages still in the Hierarchy making their way Upstream.
2. The Root Complex, on behalf of system software, issues a Configuration Read Request targeting one of the PME requester's PMCSR (e.g., reading its PME_Status bit).
3. The corresponding split completion Packet is required, as per producer/consumer ordering rules, to push all previously posted PM_PME Messages ahead of it, which in this case are PM_PME Messages that have no place to go.
4. The PME service routine cannot make progress; the PM_PME Message storage situation does not improve.
5. Deadlock occurs.

Precluding potential deadlocks requires the Root Complex to always enable forward progress under these circumstances. This must be done by accepting any PM_PME Messages that posted queue flow control credits allow for, and discarding any PM_PME Messages that create an overflow condition. This required behavior ensures that no deadlock will occur in these cases; however, PM_PME Messages will be discarded and hence lost in the process.

To ensure that no PM_PME Messages are lost permanently, all agents that are capable of generating PM_PME must implement a PME Service Timeout mechanism to ensure that their PME requests are serviced in a reasonable amount of time.

If after 100 ms (+ 50%/- 5%), the PME_Status bit of a requesting agent has not yet been cleared, the PME Service Timeout mechanism expires triggering the PME requesting agent to re-send the temporarily lost PM_PME Message. If at this time the Link is in a non-communicating state, then, prior to re-sending the PM_PME Message, the agent must reactivate the Link as defined in Section 5.3.3.2.

5.3.3.4. PME Rules

- ❑ All device Functions must implement the PCI-PM Power Management Capabilities (PMC) register and the PMCSR in accordance with the PCI-PM specification. These registers reside in the PCI-PM compliant PCI Capability List format.

PME capable Functions must implement the PME_Status bit, and underlying functional behavior, in their PMCSR.

When a Function initiates Link wakeup, or issues a PM_PME Message, it must set its PME_Status bit.

- ❑ Switches must route a PM_PME received on any Downstream Port to their Upstream Port
- ❑ On receiving a PME_Turn_Off Message, the device must block the transmission of PM_PME Messages and transmit a PME_TO_Ack Message Upstream. The component is permitted to send a PM_PME Message after the Link is returned to an L0 state through LDn.
- ❑ Before a Link or a portion of a Hierarchy is transferred into a non-communicating state (i.e., a state from which it cannot issue a PM_PME Message), a PME_Turn_Off Message must be broadcast Downstream.

5.3.3.5. PM_PME Delivery State Machine

The following diagram conceptually outlines the PM_PME delivery control state machine. This state machine determines the ability of a Link to service PME events by issuing PM_PME immediately vs. requiring Link wakeup.

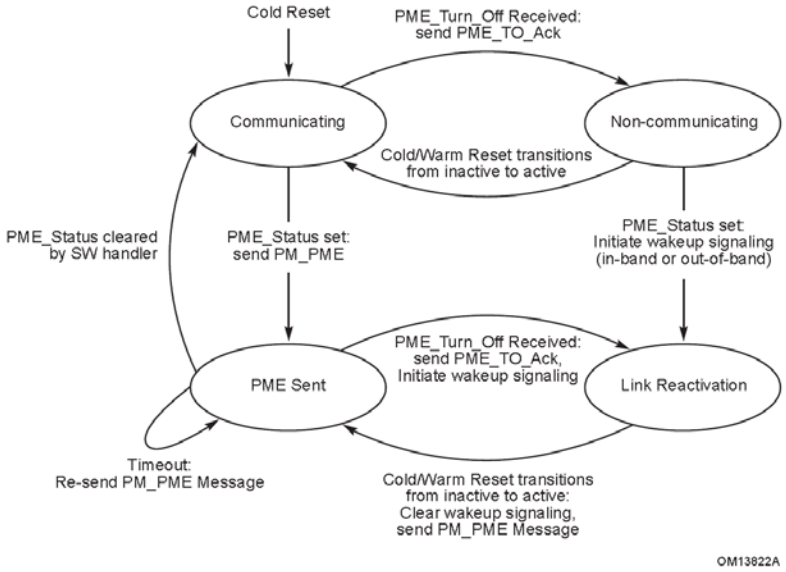


Figure 5-5: A Conceptual PME Control State Machine

Communicating State:

At initial power-up and associated reset, the Upstream Link enters the Communicating state

- If PME_Status is asserted (assuming PME delivery is enabled), a PM_PME Message will be issued Upstream, terminating at the root of the Hierarchy. The next state is the PME Sent state
- If a PME_Turn_Off Message is received, the Link enters the Non-communicating state following its acknowledgment of the Message and subsequent entry into the L2/L3 Ready state.

Non-communicating State:

- Following the restoration of power and clock, and the associated reset, the next state is the Communicating state.
- If PME_Status is asserted, the Link will transition to the Link Reactivation state, and activate the wakeup mechanism.

PME Sent State

- If PME_Status is cleared, the Function becomes PME Capable again. Next state is the Communicating state.
- If the PME_Status bit is not Clear by the time the PME service timeout expires, a PM_PME Message is re-sent Upstream. Refer to Section 5.3.3.3.1 for an explanation of the timeout mechanism.
- If a PME Message has been issued but the PME_Status has not been cleared by software when the Link is about to be transitioned into a messaging incapable state (a PME_Turn_Off Message is received), the Link transitions into Link Reactivation state after sending a PME_TO_Ack Message. The device also activates the wakeup mechanism.

Link Reactivation State

- Following the restoration of power and clock, and the associated reset, the Link resumes a transaction-capable state. The device clears the wakeup signaling, if necessary, and issues a PM_PME Upstream and transitions into the PME Sent state.

5.4. Native PCI Express Power Management Mechanisms

The following sections define power management features that require new software. While the presence of these features in new PCI Express designs will not break legacy software compatibility, taking the full advantage of them requires new code to manage them.

These features are enumerated and configured using PCI Express native configuration mechanisms as described in Chapter 7 of this specification. Refer to Chapter 7 for specific register locations, bit assignments, and access mechanisms associated with these PCI Express-PM features.

5.4.1. Active State Power Management (ASPM)

All Ports not associated with an Internal Root Complex Link or system Egress Port are required to support the minimum requirements defined herein for Active State Link PM. This feature must be treated as being orthogonal to the PCI-PM software compatible features from a minimum requirements perspective. For example, the Root Complex is exempt from the PCI-PM software compatible features requirements; however, it must implement the minimum requirements of ASPM.

Components in the D0 state (i.e., fully active state) normally keep their Upstream Link in the active L0 state, as defined in Section 5.3.2. ASPM defines a protocol for components in the D0 state to reduce Link power by placing their Links into a low power state and instructing the other end of the Link to do likewise. This capability allows hardware-autonomous, dynamic Link power reduction beyond what is achievable by software-only controlled (i.e., PCI-PM software driven) power management.

Two low power "standby" Link states are defined for ASPM. The L0s low power Link state is optimized for short entry and exit latencies, while providing substantial power savings. If the L0s state is enabled in a device, it is recommended that the device bring its Transmit Link into the L0s

state whenever that Link is not in use (refer to Section 5.4.1.1.1 for details relating to the L0s invocation policy). Component support of the L0s Link state from within the D0 device state is optional unless the applicable form factor specification for the Link explicitly requires it.

The L1 Link state is optimized for maximum power savings at a cost of longer entry and exit latencies. L1 reduces Link power beyond the L0s state for cases where very low power is required and longer transition times are acceptable. ASPM support for the L1 Link state is optional unless specifically required by a particular form factor.

Optional L1 PM Substates L1.1 and L1.2 are defined. These substates can further reduce Link power for cases where very low idle power is required, and longer transition times are acceptable.

Comment [VS175]: ECN: L1 PM Substates with CLKREQ

Each component must report its level of support for ASPM in the ASPM Support field. As applicable, each component shall also report its L0s and L1 exit latency (the time that it requires to transition from the L0s or L1 state to the L0 state). Endpoint Functions must also report the worst-case latency that they can withstand before risking, for example, internal FIFO overruns due to the transition latency from L0s or L1 to the L0 state. Power management software can use the provided information to then enable the appropriate level of ASPM.

The L0s exit latency may differ significantly if the reference clock for opposing sides of a given Link is provided from the same source, or delivered to each component from a different source. PCI Express-PM software informs each device of its clock configuration via the Common Clock Configuration bit in its Capability structure's Link Control register. This bit serves as the determining factor in the L0s exit latency value reported by the device. ASPM may be enabled or disabled by default depending on implementation specific criteria and/or the requirements of the associated form factor specification(s). Software can enable or disable ASPM using a process described in Section 5.4.1.3.1.

Power management software enables or disables ASPM in each Port of a component by programming the ASPM Control field. Note that new BIOS code can effectively enable or disable ASPM functionality when running with a legacy operating system, but a PCI Express-aware operating system might choose to override ASPM settings configured by the BIOS.



IMPLEMENTATION NOTE

Isochronous Traffic and ASPM

Isochronous traffic requires bounded service latency. ASPM may add latency to isochronous transactions beyond expected limits. A possible solution would be to disable ASPM for devices that are configured with an Isochronous Virtual Channel.

For ARI Devices, ASPM Control is determined solely by the setting in Function 0, regardless of Function 0's D-state. The ASPM Control settings in other Functions are ignored by the component.

An Upstream Port of a non-ARI multi-Function device may be programmed with different values in their respective ASPM Control fields of each Function. The policy for such a component will be dictated by the most active common denominator among all D0 Functions according to the following rules:

- Functions in a non-D0 state (D1 and deeper) are ignored in determining the ASPM policy
- If any of the Functions in the D0 state has its ASPM disabled (ASPM Control field = 00b) or if at least one of the Functions in the D0 state is enabled for L0s only (ASPM Control field = 01b) and at least one other Function in the D0 state is enabled for L1 only (ASPM Control field = 10b), then ASPM is disabled for the entire component
- Else, if at least one of the Functions in the D0 state is enabled for L0s only (ASPM Control field = 01b), then ASPM is enabled for L0s only
- Else, if at least one of the Functions in the D0 state is enabled for L1 only (ASPM Control field = 10b), then ASPM is enabled for L1 only
- Else, ASPM is enabled for both L0s and L1 states

Note that the components must be capable of changing their behavior during runtime as device Functions enter and exit low power device states. For example, if one Function within a multi-Function device is programmed to disable ASPM, then ASPM must be disabled for that device while that Function is in the D0 state. Once the Function transitions to a non-D0 state, ASPM can be enabled if all other Functions are enabled for ASPM.

5.4.1.1. L0s ASPM State

Device support of the L0s low power Link state is optional unless the applicable form factor specification for the Link explicitly requires it.



IMPLEMENTATION NOTE

Potential Issues With Legacy Software When L0s is Not Supported

In earlier versions of this specification, device support of L0s was mandatory, and software could legitimately assume that all devices support L0s. Newer hardware components that do not support L0s may encounter issues with such "legacy software". Such software might not even check the ASPM Support field in the Link Capability register, might not recognize the subsequently defined values (00b and 10b) for the ASPM Support field, or might not follow the policy of enabling L0s only if components on both sides of the Link each support L0s.

Legacy software (either operating system or firmware) that encounters the previously reserved value 00b (No ASPM Support), will most likely refrain from enabling L1, which is intended behavior. Legacy software will also most likely refrain from enabling L0s for that component's Transmitter (also intended behavior), but it is unclear if such software will also refrain from enabling L0s for the component on the other side of the Link. If software enables L0s on one side when the component on the other side does not indicate that it supports L0s, the result is undefined. Situations where the resulting behavior is unacceptable may need to be handled by updating the legacy software, resorting to "blacklists" or similar mechanisms directing the legacy software not to enable L0s, or simply not supporting the problematic system configurations.

On some platforms, firmware controls ASPM, and the operating system may either preserve or override the ASPM settings established by firmware. This will be influenced by whether the operating system supports controlling ASPM, and in some cases by whether the firmware permits the operating system to take control of ASPM. Also, ASPM control with hotplug operations may be influenced by whether native PCI Express hotplug versus ACPI hotplug is used. Addressing any legacy software issues with L0s may require updating the firmware, the operating system, or both.

When a component does not advertise that it supports L0s, as indicated by its ASPM Support field value being 00b or 10b, it is recommended that the component's L0s Exit Latency field return a value of 111b, indicating the maximum latency range. Advertising this maximum latency range may help discourage legacy software from enabling L0s if it otherwise would do so, and thus may help avoid problems caused by legacy software mistakenly enabling L0s on this component or the component on the other side of the Link.

Transaction Layer and Link Layer timers are not affected by a transition to the L0s state (i.e., they must follow the rules as defined in their respective chapters).

IMPLEMENTATION NOTE

Minimizing L0s Exit Latency

L0s exit latency depends mainly on the ability of the Receiver to quickly acquire bit and Symbol synchronization. Different approaches exist for high-frequency clocking solutions which may differ significantly in their L0s exit latency, and therefore in the efficiency of ASPM. To achieve maximum power savings efficiency with ASPM, L0s exit latency should be kept low by proper selection of the clocking solution.

5.4.1.1.1. Entry into the L0s State

Entry into the L0s state is managed separately for each direction of the Link. It is the responsibility of each device at either end of the Link to initiate an entry into the L0s state on its transmitting Lanes. Software must not enable L0s in either direction on a given Link unless components on both sides of the Link each support L0s; otherwise, the result is undefined.

A Port that is disabled for the L0s state must not transition its transmitting Lanes to the L0s state. However, if the Port advertises that it supports L0s, Port must be able to tolerate having its Receiver Port Lanes enter L0s, (as a result of the device at the other end bringing its transmitting Lanes into L0s state), and then later returning to the L0 state.

L0s Invocation Policy

Ports that are enabled for L0s entry generally should transition their Transmit Lanes to the L0s state if the defined idle conditions (below) are met for a period of time, recommended not to exceed 7 µs. Within this time period, the policy used by the Port to determine when to enter L0s is implementation specific. It is never mandatory for a Transmitter to enter L0s.

Definition of Idle

The definition of an "idle" Upstream Port varies with device Function category. An Upstream Port of a multi-Function device is considered idle only when all of its Functions are idle.

A non-Switch Port is determined to be idle if the following conditions are met:

- No TLP is pending to transmit over the Link, or no FC credits are available to transmit any TLPs
- No DLLPs are pending for transmission

A Switch Upstream Port Function is determined to be idle if the following conditions are met:

- None of the Switch's Downstream Port Receive Lanes are in the L0, Recovery, or Configuration state
- No pending TLPs to transmit, or no FC credits are available to transmit anything
- No DLLPs are pending for transmission

A Switch's Downstream Port is determined to be idle if the following conditions are met:

- The Switch's Upstream Port's Receive Lanes are not in the L0, Recovery, or Configuration state
- No pending TLPs to transmit on this Link, or no FC credits are available
- No DLLPs are pending for transmission

Refer to Section 4.2 for details on L0s entry by the Physical Layer.

5.4.1.1.2. Exit from the L0s State

A component with its Transmitter in L0s must initiate L0s exit when it has a TLP or DLLP to transmit across the Link. Note that a transition from the L0s Link state does not depend on the status (or availability) of FC credits. The Link must be able to reach the L0 state, and to exchange FC credits across the Link. For example, if all credits of some type were consumed when the Link entered L0s, then any component on either side of the Link must still be able to transition the Link to the L0 state when new credits need to be sent across the Link. Note that it may be appropriate for a component to anticipate the end of the idle condition and initiate L0s transmit exit; for example, when a NP request is received.

Downstream Initiated Exit

The Upstream Port of a component is permitted to initiate an exit from the L0s low-power state on its Transmit Link, (Upstream Port Transmit Lanes in the case of a Downstream Switch), if it needs to communicate through the Link. The component initiates a transition to the L0 state on Lanes in the Upstream direction as described in Section 4.2.

If the Upstream component is a Switch (i.e., it is not the Root Complex), then it must initiate a transition on its Upstream Port Transmit Lanes (if the Upstream Port's Transmit Lanes are in a low-power state) as soon as it detects an exit from L0s on any of its Downstream Ports.

Upstream Initiated Exit

A Downstream Port is permitted to initiate an exit from L0s low power state on any of its Transmit Links if it needs to communicate through the Link. The component initiates a transition to the L0 state on Lanes in the Downstream direction as described in Chapter 4.

If the Downstream component contains a Switch, it must initiate a transition on all of its Downstream Port Transmit Lanes that are in L0s at that time as soon as it detects an exit from L0s on its Upstream Port. Links that are already in the L0 state are not affected by this transition. Links whose Downstream component is in a low-power state (i.e., D1-$D3_{hot}$ states) are also not affected by the exit transitions.

For example, consider a Switch with an Upstream Port in L0s and a Downstream device in a D1 state. A configuration request packet travels Downstream to the Switch, intending ultimately to reprogram the Downstream device from D1 to D0. The Switch's Upstream Port Link must transition to the L0 state to allow the packet to reach the Switch. The Downstream Link connecting to the device in D1 state will not transition to the L0 state yet; it will remain in the L1 state. The captured packet is checked and routed to the Downstream Port that shares a Link with the Downstream device that is in D1. As described in Section 4.2, the Switch now transitions the Downstream Link to the L0 state. Note that the transition to the L0 state was triggered by the packet being routed to that particular Downstream L1 Link, and not by the transition of the

Upstream Port's Link to the L0 state. If the packet's destination was targeting a different Downstream Link, then that particular Downstream Link would have remained in the L1 state.

5.4.1.2. L1 ASPM State

A component may optionally support the ASPM L1 state; a state that provides greater power savings at the expense of longer exit latency. L1 exit latency is visible to software, and reported via the L1 Exit Latency field.



IMPLEMENTATION NOTE

Potential Issues With Legacy Software When Only L1 is Supported

In earlier versions of this specification, device support of L0s was mandatory, and there was no architected ASPM Support field value to indicate L1 support without L0s support. Newer hardware components that support only L1 may encounter issues with "legacy software", i.e., software that does not recognize the subsequently defined value for the ASPM Support field.

Legacy software that encounters the previously reserved value 10b (L1 Support), may refrain from enabling both L0s and L1, which unfortunately avoids using L1 with new components that support only L1. While this may result in additional power being consumed, it should not cause any functional misbehavior. However, the same issues with respect to legacy software enabling L0s exist for this 10b case as are described in the Implementation Note "Potential Issues With Legacy Software When L0s is Not Supported" in Section 5.4.1.1.

When supported, L1 entry is disabled by default in the ASPM Control field. Software must enable ASPM L1 on the Downstream component only if it is supported by both components on a Link. Software must sequence the enabling and disabling of ASPM L1 such that the Upstream component is enabled before the Downstream component and disabled after the Downstream component.

5.4.1.2.1. Entry into the L1 State

An Upstream Port on a component enabled for L1 ASPM entry may initiate entry into the L1 Link state.

See Section 5.5.1 for details on transitions into either the L1.1 or L1.2 substates.

Comment [VS176]: ECN: L1 PM Substates with CLKREQ



IMPLEMENTATION NOTE

Initiating L1

This specification does not dictate when a component with an Upstream Port must initiate a transition to the L1 state. The interoperable mechanisms for transitioning into and out of L1 are defined within this specification; however, the specific ASPM policy governing when to transition into L1 is left to the implementer.

One possible approach would be for the Downstream device to initiate a transition to the L1 state once the device has both its Receiver and Transmitter in the L0s state (RxL0s and TxL0s) for a set amount of time. Another approach would be for the Downstream device to initiate a transition to the L1 state once the Link has been idle in L0 for a set amount of time. This is particularly useful if L0s entry is not enabled. Still another approach would be for the Downstream device to initiate a transition to the L1 state if it has completed its assigned tasks. Note that a component's L1 invocation policy is in no way limited by these few examples.

Three power management Messages provide support for the ASPM L1 state:

- PM_ Active_State_Request_L1 (DLLP)
- PM_ Request_Ack (DLLP)
- PM_Active_State_Nak (TLP)

Downstream components enabled for ASPM L1 entry negotiate for L1 entry with the Upstream component on the Link.

A Downstream Port must accept a request to enter L1 if all of the following conditions are true:

- The Port supports ASPM L1 entry, and ASPM L1 entry is enabled.[67]
- No TLP is scheduled for transmission
- No Ack or Nak DLLP is scheduled for transmission

[67] Software must enable ASPM L1 for the Downstream component only if it is also enabled for the Upstream component.

A Switch Upstream Port may request L1 entry on its Link provided all of the following conditions are true:

- The Upstream Port supports ASPM L1 entry and it is enabled
- All of the Switch's Downstream Port Links are in the L1 state (or deeper)
- No pending TLPs to transmit
- No pending DLLPs to transmit
- The Upstream Port's Receiver is idle for an implementation specific set amount of time

Note that it is legitimate for a Switch to be enabled for the ASPM L1 Link state on any of its Downstream Ports and to be disabled or not even supportive of ASPM L1 on its Upstream Port. In that case, Downstream Ports may enter the L1 Link state, but the Switch will never initiate an ASPM L1 entry transition on its Upstream Port.

ASPM L1 Negotiation Rules (see Figure 5-6 and Figure 5-7)

- The Downstream component must not initiate ASPM L1 entry until it accumulates at least the minimum number of credits required to send the largest possible packet for any FC type.
- Upon deciding to enter a low-power Link state, the Downstream component must block movement of all TLPs from the Transaction Layer to the Data Link Layer for transmission (including completion packets).

If any TLPs become available from the Transaction Layer for transmission during the L1 negotiation process, the transition to L1 must first be completed and then the Downstream component must initiate a return to L0. Refer to Section 5.2 if the negotiation to L1 is interrupted.

- The Downstream component must wait until it receives a Link Layer acknowledgement for the last TLP it had previously sent (i.e., the retry buffer is empty). The component must retransmit a TLP out of its Data Link Layer Retry buffer if required by the Data Link Layer rules.
- The Downstream component then initiates ASPM negotiation by sending a PM_Active_State_Request_L1 DLLP onto its Transmit Lanes. The Downstream component sends this DLLP repeatedly with no more than four (when using 8b/10b encoding) or 16 (when using 128b/130b encoding) Symbol times of idle between subsequent transmissions of the PM_Active_State_Request_L1 DLLP. The transmission of other DLLPs and SKP Ordered Sets must occur as required at any time between PM_Active_State_Request_L1 transmissions, and do not contribute to this idle time limit. Transmission of SKP Ordered Sets during L1 entry follows the clock tolerance compensation rules in Section 4.2.7.

- The Downstream component continues to transmit the PM_Active_State_Request_L1 DLLP as described above until it receives a response from the Upstream device (see below). The Downstream component remains in this loop waiting for a response from the Upstream component.

During this waiting period, the Downstream component must not initiate any Transaction Layer transfers. It must still accept TLPs and DLLPs from the Upstream component, storing for later transmission any TLP responses required. It continues to respond with DLLPs, including FC update DLLPs, as needed by the Link Layer protocol.

If the Downstream component for any reason needs to transmit a TLP on the Link, it must first complete the transition to the low-power Link state. Once in a lower power Link state, the Downstream component must then initiate exit of the low-power Link state to handle the transfer. Refer to Section 5.2 if the negotiation to L1 is interrupted.

- The Upstream component must immediately (while obeying all other rules in this specification) respond to the request with either an acceptance or a rejection of the request.

If the Upstream component is not able to accept the request, it must immediately (while obeying all other rules in this specification) reject the request.

- Refer to Section 5.2 if the negotiation to L1 is interrupted.

Rules in case of rejection:

- In the case of a rejection, the Upstream component must schedule, as soon as possible, a rejection by sending the PM_Active_State_Nak Message to the Downstream component. Once the PM_Active_State_Nak Message is sent, the Upstream component is permitted to initiate any TLP or DLLP transfers.
- If the request was rejected, it is generally recommended that the Downstream component immediately transition its Transmit Lanes into the L0s state, provided L0s is enabled and that conditions for L0s entry are met.
- Prior to transmitting a PM_Active_State_Request_L1 DLLP associated with a subsequent ASPM L1 negotiation sequence, the Downstream component must either enter and exit L0s on its Transmitter, or it must wait at least 10 µs from the last transmission of the PM_Active_State_Request_L1 DLLP associated with the preceding ASPM L1 negotiation. This 10 µs timer must count only time spent in the LTSSM L0 and L0s states. The timer must hold in the LTSSM Recovery state. If the Link goes down and comes back up, the timer is ignored and the component is permitted to issue new ASPM L1 request after the Link has come back up.

IMPLEMENTATION NOTE

ASPM L1 Accept/Reject Considerations for the Upstream Component

When the Upstream component has responded to the Downstream component's ASPM L1 request with a PM_Request_Ack DLLP to accept the L1 entry request, the ASPM L1 negotiation protocol clearly and unambiguously ends with the Link entering L1. However, if the Upstream component responds with a PM_Active_State_Nak Message to reject the L1 entry request, the termination of the ASPM L1 negotiation protocol is less clear. Therefore, both components need to be designed to unambiguously terminate the protocol exchange. If this is not done, there is the risk that the two components will get out of sync with each other, and the results may be undefined. For example, consider the following case:

- The Downstream component requests ASPM L1 entry by transmitting a sequence of PM_Active_State_Request_L1 DLLPs.
- Due to a temporary condition, the Upstream component responds with a PM_Active_State_Nak Message to reject the L1 request.
- The Downstream component continues to transmit the PM_Active_State_Request_L1 DLLPs for some time before it is able to respond to the PM_Active_State_Nak Message.
- Meanwhile, the temporary condition that previously caused the Upstream component to reject the L1 request is resolved, and the Upstream component erroneously sees the continuing PM_Active_State_Request_L1 DLLPs as a new request to enter L1, and responds by transmitting PM_Request_Ack DLLPs Downstream.

At this point, the result is undefined, because the Downstream component views the L1 request as rejected and finishing, but the Upstream component views the situation as a second L1 request being accepted.

To avoid this situation, the Downstream component needs to provide a mechanism to distinguish between one ASPM L1 request and another. The Downstream component does this by entering L0s or by waiting a minimum of 10 µs from the transmission of the last PM_Active_State_Request_L1 DLLP associated with the first ASPM L1 request before starting transmission of the PM_Active_State_Request_L1 DLLPs associated with the second request (as described above).

If the Upstream component is capable of exhibiting the behavior described above, then it is necessary for the Upstream component to recognize the end of an L1 request sequence by detecting a transition to L0s on its Receiver or a break in the reception of PM_Active_State_Request_L1 DLLPs of 9.5 µs measured while in L0/L0s or more as a separation between ASPM L1 requests by the Downstream component.

If there is a possibility of ambiguity, the Upstream component should reject the L1 request to avoid potentially creating the ambiguous situation outlined above.

Rules in case of acceptance:

- If the Upstream component is ready to accept the request, it must block scheduling of any TLPs from the Transaction Layer.
- The Upstream component then must wait until it receives a Data Link Layer acknowledgement for the last TLP it had previously sent. The Upstream component must retransmit a TLP if required by the Data Link Layer rules.
- Once all TLPs have been acknowledged, the Upstream component sends a PM_Request_Ack DLLP Downstream. The Upstream component sends this DLLP repeatedly with no more than four (when using 8b/10b encoding) Symbol times of idle between subsequent transmissions of the PM_Request_Ack DLLP. The transmission of SKP Ordered Sets must occur as required at any time between PM_Request_Ack transmissions, and do not contribute to this idle time limit. Transmission of SKP Ordered Sets during L1 entry follows the clock tolerance compensation rules in Section 4.2.7.
- The Upstream component continues to transmit the PM_Request_Ack DLLP as described above until it observes its Receive Lanes enter into the Electrical Idle state. Refer to Chapter 4 for more details on the Physical Layer behavior.
- If the Upstream component needs, for any reason, to transmit a TLP on the Link after it sends a PM_Request_Ack DLLP, it must first complete the transition to the low-power state, and then initiate an exit from the low-power state to handle the transfer once the Link is back to L0. Refer to Section 5.2 if the negotiation to L1 is interrupted.
 - The Upstream component must initiate an exit from L1 in this case even if it does not have the required flow control credit to transmit the TLP(s).
- When the Downstream component detects a PM_Request_Ack DLLP on its Receive Lanes (signaling that the Upstream device acknowledged the transition to L1 request), the Downstream component then ceases sending the PM_Active_State_Request_L1 DLLP, disables DLLP, TLP transmission and brings its Transmit Lanes into the Electrical Idle state.
- When the Upstream component detects an Electrical Idle on its Receive Lanes (signaling that the Downstream component has entered the L1 state), it then ceases to send the PM_Request_Ack DLLP, disables DLLP, TLP transmission and brings the Downstream direction of the Link into the Electrical Idle state.

Notes:

1. The transaction Layer Completion Timeout mechanism is not affected by transition to the L1 state (i.e., it must keep counting).
2. Flow Control Update timers are frozen while the Link is in L1 state to prevent a timer expiration that will unnecessarily transition the Link back to the L0 state.

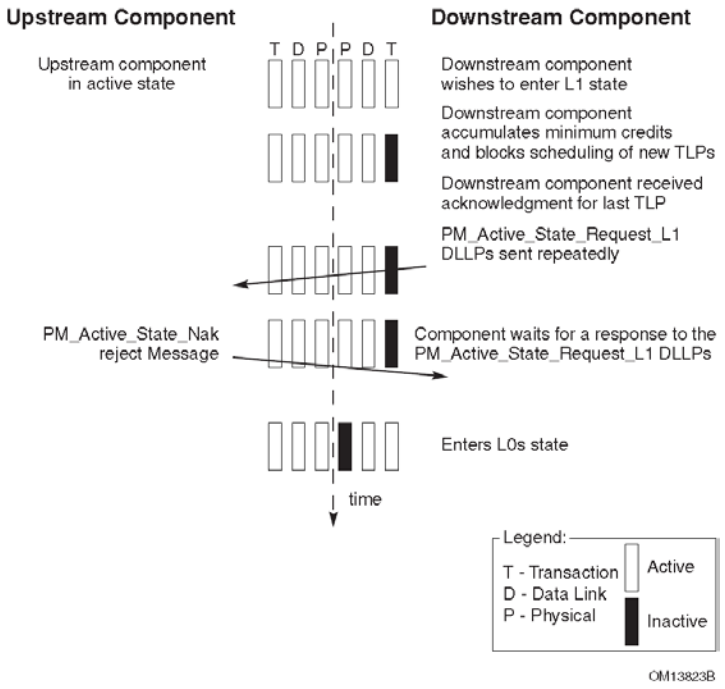


Figure 5-6: L1 Transition Sequence Ending with a Rejection (L0s Enabled)

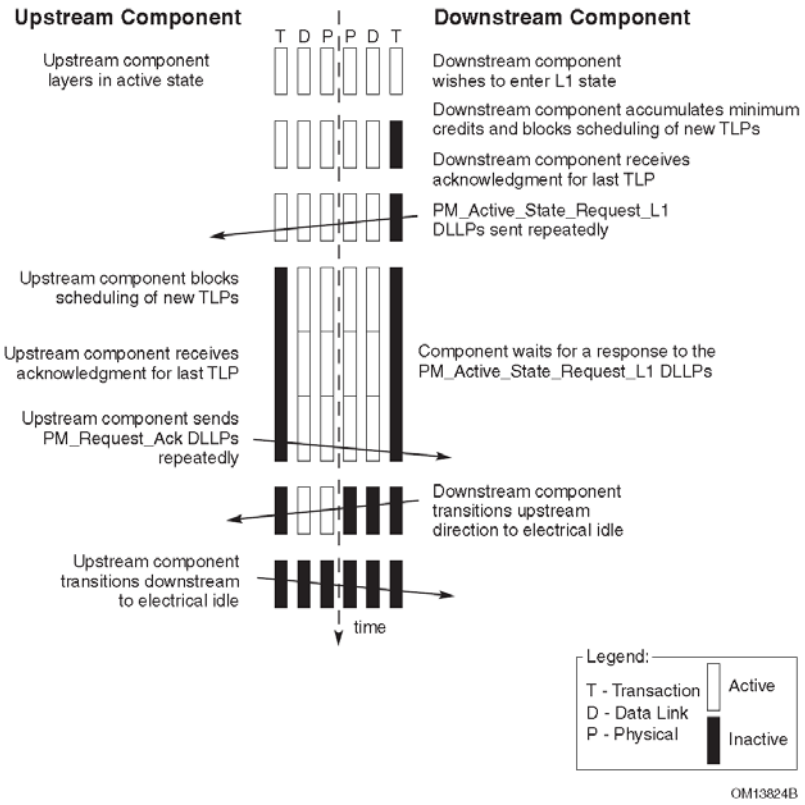


Figure 5-7: L1 Successful Transition Sequence

5.4.1.2.2. Exit from the L1 State

Components on either end of a Link may initiate an exit from the L1 Link state.

See Section 5.5.1 for details on transitions into either the L1.1 or L1.2 substates.

Comment [VS177]: ECN: L1 PM Substates with CLKREQ

Upon exit from L1, it is recommended that the Downstream component send flow control update DLLPs for all enabled VCs and FC types starting within 1 µs of L1 exit.

Downstream Component Initiated Exit

An Upstream Port must initiate an exit from L1 on its Transmit Lanes if it needs to communicate through the Link. The component initiates a transition to the L0 state as described in Chapter 4. The Upstream component must respond by initiating a similar transition of its Transmit Lanes.

If the Upstream component is a Switch Downstream Port, (i.e., it is not a Root Complex Root Port), the Switch must initiate an L1 exit transition on its Upstream Port's Transmit Lanes, (if the Upstream Port's Link is in the L1 state), as soon as it detects the L1 exit activity on any of its

Downstream Port Links. Since L1 exit latencies are relatively long, a Switch must not wait until its Downstream Port Link has fully exited to L0 before initiating an L1 exit transition on its Upstream Port Link. Waiting until the Downstream Link has completed the L0 transition will cause a Message traveling through several Switches to experience accumulating latency as it traverses each Switch.

A Switch is required to initiate an L1 exit transition on its Upstream Port Link after no more than 1 µs from the beginning of an L1 exit transition on any of its Downstream Port Links. Refer to Section 4.2 for details of the Physical Layer signaling during L1 exit.

Consider the example in Figure 5-8. The numbers attached to each Port represent the corresponding Port's reported Transmit Lanes L1 exit latency in units of microseconds.

Links 1, 2, and 3 are all in the L1 state, and Endpoint C initiates a transition to the L0 state at time T. Since Switch B takes 32 µs to exit L1 on its Ports, Link 3 will transition to the L0 state at T+32 (longest time considering T+8 for the Endpoint C, and T+32 for Switch B).

Switch B is required to initiate a transition from the L1 state on its Upstream Port Link (Link 2) after no more than 1 µs from the beginning of the transition from the L1 state on Link 3. Therefore, transition to the L0 state will begin on Link 2 at T+1. Similarly, Link 1 will start its transition to the L0 state at time T+2.

Following along as above, Link 2 will complete its transition to the L0 state at time T+33 (since Switch B takes longer to transition and it started at time T+1). Link 1 will complete its transition to the L0 state at time T+34 (since the Root Complex takes 32 µs to transition and it started at time T+2).

Therefore, among Links 1, 2, and 3, the Link to complete the transition to the L0 state last is Link 1 with a 34 µs delay. This is the delay experienced by the packet that initiated the transition in Endpoint C.

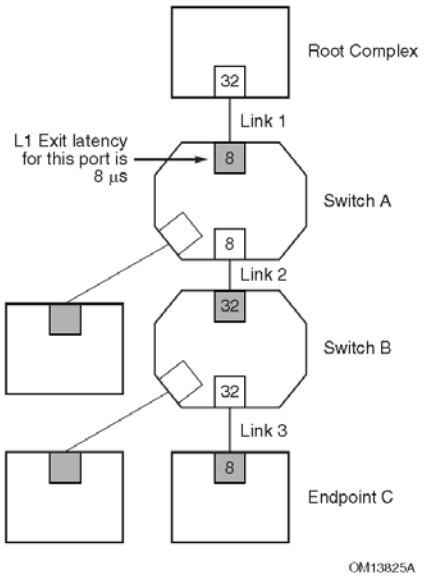


Figure 5-8: Example of L1 Exit Latency Computation

Switches are not required to initiate an L1 exit transition on any other of their Downstream Port Links.

Upstream Component Initiated Exit

A Root Complex, or a Switch must initiate an exit from L1 on any of its Root Ports, or Downstream Port Links if it needs to communicate through that Link. The Switch or Root Complex must be capable of initiating L1 exit even if it does not have the flow control credits needed to transmit a given TLP. The component initiates a transition to the L0 state as described in Chapter 4. The Downstream component must respond by initiating a similar transition on its Transmit Lanes.

If the Downstream component contains a Switch, it must initiate a transition on all of its Downstream Links (assuming the Downstream Link is in an ASPM L1 state) as soon as it detects an exit from L1 state on its Upstream Port Link. Since L1 exit latencies are relatively long, a Switch must not wait until its Upstream Port Link has fully exited to L0 before initiating an L1 exit transition on its Downstream Port Links. If that were the case, a Message traveling through multiple Switches would experience accumulating latency as it traverses each Switch.

A Switch is required to initiate a transition from L1 state on all of its Downstream Port Links that are currently in L1 after no more than 1 µs from the beginning of a transition from L1 state on its Upstream Port. Refer to Section 4.2 for details of the Physical Layer signaling during L1 exit. Downstream Port Links that are already in the L0 state do not participate in the exit transition. Downstream Port Links whose Downstream component is in a low power D-state (D1-$D3_{hot}$) are also not affected by the L1 exit transitions (i.e., such Links must not be transitioned to the L0 state).

5.4.1.3. ASPM Configuration

All Functions must implement the following configuration bits in support of ASPM. Refer to Chapter 7 for configuration register assignment and access mechanisms.

Each component reports its level of support for ASPM in the ASPM Support field below.

Table 5-3: Encoding of the ASPM Support Field

Field	Description
ASPM Support	00b – No ASPM support 01b – L0s supported 10b – L1 supported 11b – L0s and L1 supported

Software must not enable L0s in either direction on a given Link unless components on both sides of the Link each support L0s; otherwise, the result is undefined.

Each component reports the source of its reference clock in its Slot Clock Configuration bit located in its Capability structure's Link Status register.

Table 5-4: Description of the Slot Clock Configuration Bit

Bit	Description
Slot Clock Configuration	This bit, when set, indicates that the component uses the same physical reference clock that the platform provides on the connector. If the component uses an independent clock irrespective of the presence of a reference on the connector, this bit must be clear. For Root and Switch Downstream Ports, this bit, when set, indicates that the Downstream Port is using the same reference clock as the Downstream component or the slot. For Switch and Bridge Upstream Ports, this bit when set, indicates that the Upstream Port is using the same reference clock that the platform provides. Otherwise it is clear.

Each component must support the Common Clock Configuration bit in their Capability structure's Link Control register. Software writes to this register bit to indicate to the device whether it is sharing the same clock source as the device on the other end of the Link.

Table 5-5: Description of the Common Clock Configuration Bit

Bit	Description
Common Clock Configuration	This bit when set indicates that this component and the component at the opposite end of the Link are operating with a common clock source. A value of 0b indicates that this component and the component at the opposite end of the Link are operating with separate reference clock sources. Default value of this bit is 0b. Components utilize this common clock configuration information to report the correct L0s and L1 Exit Latencies.

Each Port reports the L0s and L1 exit latency (the time that they require to transition their Receive Lanes from the L0s or L1 state to the L0 state) in the L0s Exit Latency and the L1 Exit Latency configuration fields, respectively. If a Port does not support L0s or ASPM L1, the value of the respective exit latency field is undefined.

Table 5-6: Encoding of the L0s Exit Latency Field

Field	Description
L0s Exit Latency	000b – Less than 64 ns 001b – 64 ns to less than 128 ns 010b – 128 ns to less than 256 ns 011b – 256 ns to less than 512 ns 100b – 512 ns to less than 1 µs 101b – 1 µs to less than 2 µs 110b – 2 µs to 4 µs 111b – More than 4 µs

Table 5-7: Encoding of the L1 Exit Latency Field

Field	Description
L1 Exit Latency	000b – Less than 1 µs 001b – 1 µs to less than 2 µs 010b – 2 µs to less than 4 µs 011b – 4 µs to less than 8 µs 100b – 8 µs to less than 16 µs 101b – 16 µs to less than 32 µs 110b – 32 µs to 64 µs 111b – More than 64 µs

Endpoints also report the additional latency that they can absorb due to the transition from L0s state or L1 state to the L0 state. This is reported in the Endpoint L0s Acceptable Latency and Endpoint L1 Acceptable Latency fields, respectively.

Power management software, using the latency information reported by all components in the Hierarchy, can enable the appropriate level of ASPM by comparing exit latency for each given path from Root to Endpoint against the acceptable latency that each corresponding Endpoint can withstand.

Table 5-8: Encoding of the Endpoint L0s Acceptable Latency Field

Field	Description
Endpoint L0s Acceptable Latency	000b – Maximum of 64 ns 001b – Maximum of 128 ns 010b – Maximum of 256 ns 011b – Maximum of 512 ns 100b – Maximum of 1 µs 101b – Maximum of 2 µs 110b – Maximum of 4 µs 111b – No limit

Table 5-9: Encoding of the Endpoint L1 Acceptable Latency Field

Field	Description
Endpoint L1 Acceptable Latency	000b – Maximum of 1 µs 001b – Maximum of 2 µs 010b – Maximum of 4 µs 011b – Maximum of 8 µs 100b – Maximum of 16 µs 101b – Maximum of 32 µs 110b – Maximum of 64 µs 111b – No limit

Power management software enables or disables ASPM in each component by programming the ASPM Control field.

Table 5-10: Encoding of the ASPM Control Field

Field	Description
ASPM Control	00b – Disabled 01b – L0s Entry Enabled 10b – L1 Entry Enabled 11b – L0s and L1 Entry enabled

ASPM Control = 00b

Port's Transmitter must not enter L0s.

Ports connected to the Downstream end of the Link must not issue a PM_Active_State_Request_L1 DLLP on its Upstream Link.

Ports connected to the Upstream end of the Link receiving a L1 request must respond with negative acknowledgement.

ASPM Control = 01b

Port must bring a Link into L0s state if all conditions are met.

Ports connected to the Downstream end of the Link must not issue a PM_Active_State_Request_L1 DLLP on its Upstream Link.

Ports connected to the Upstream end of the Link receiving a L1 request must respond with negative acknowledgement.

ASPM Control = 10b

Port's Transmitter must not enter L0s.

Ports connected to the Downstream end of the Link may issue PM_Active_State_Request_L1 DLLPs.

Ports connected to the Upstream end of the Link must respond with positive acknowledgement to a L1 request and transition into L1 if the conditions for the Root Complex Root Port or Switch Downstream Port in Section 5.4.1.2.1 are met.

ASPM Control = 11b

Port must bring a Link into the L0s state if all conditions are met.

Ports connected to the Downstream end of the Link may issue PM_Active_State_Request_L1 DLLPs.

Ports connected to the Upstream end of the Link must respond with positive acknowledgement to a L1 request and transition into L1 if the conditions for the Root Complex Root Port or Switch Downstream Port in Section 5.4.1.2.1 are met.

5.4.1.3.1. Software Flow for Enabling or Disabling ASPM

Following is an example software algorithm that highlights how to enable or disable ASPM in a component.

- PCI Express components power up with an appropriate value in their Slot Clock Configuration bit. The method by which they initialize this bit is device-specific.
- PCI Express system software scans the Slot Clock Configuration bit in the components on both ends of each Link to determine if both are using the same reference clock source or reference clocks from separate sources. If the Slot Clock Configuration bits in both devices are Set, they are both using the same reference clock source, otherwise they're not.
- PCI Express software updates the Common Clock Configuration bits in the components on both ends of each Link to indicate if those devices share the same reference clock and triggers Link retraining by writing 1b to the Retrain Link bit in the Link Control register of the Upstream component.
- Devices must reflect the appropriate L0s/L1 exit latency in their L0s/L1 Exit Latency fields, per the setting of the Common Clock Configuration bit.
- PCI Express system software then reads and calculates the L0s/L1 exit latency for each Endpoint based on the latencies reported by each Port. Refer to Section 5.4.1.2.2 for an example.
- For each component with one or more Endpoint Functions, PCI Express system software examines the Endpoint L0s/L1 Acceptable Latency, as reported by each Endpoint Function in its Link Capabilities register, and enables or disables L0s/L1 entry (via the ASPM Control field in the Link Control register) accordingly in some or all of the intervening device Ports on that hierarchy.

5.5. L1 PM Substates

Comment [VS178]: ECN: L1 PM Substates with CLKREQ

L1 PM Substates establish a Link power management regime that creates lower power substates of the L1 Link state (see Figure 5-9), and associated mechanisms for using those substates. The L1 PM Substates are:

- L1.0 substate
 - The L1.0 substate corresponds to the conventional L1 Link state. This substate is entered whenever the Link enters L1. The L1 PM Substate mechanism defines transitions from this substate to and from the L1.1 and L1.2 substates.
 - The Upstream and Downstream Ports must be enabled to detect Electrical Idle exit as required in Section 4.2.6.7.2.
- L1.1 substate
 - Link common mode voltages are maintained.
 - Uses a bidirectional open-drain clock request (CLKREQ#) signal for entry to and exit from this state.

- The Upstream and Downstream Ports are not required to be enabled to detect Electrical Idle.

- L1.2 substate
 - Link common mode voltages are not required to be maintained.
 - Uses a bidirectional open-drain clock request (CLKREQ#) signal for entry to and exit from this state.
 - The Upstream and Downstream Ports are not required to be enabled to detect electrical idle (EI).

Ports that support L1 PM Substates must not require a reference clock while in L1 PM Substates other than L1.0.

Ports that support the L1.2 substate for ASPM L1 must support Latency Tolerance Reporting (LTR).

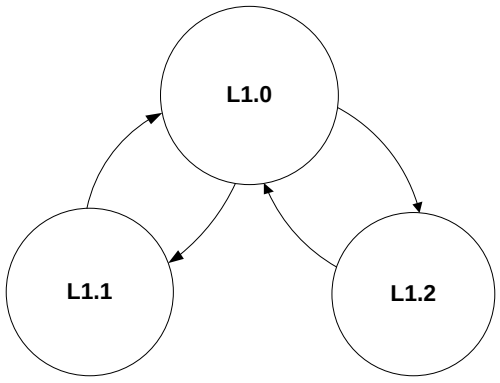


Figure 5-9: State Diagram for L1 PM Substates

When enabled, the L1 PM Substates mechanism applies the following additional requirements to the CLKREQ# signal:

- The CLKREQ# signal must be supported as a bi-directional open drain signal by both the Upstream and Downstream Ports of the Link. Each Port must have a unique instance of the signal, and the Upstream and Downstream Port CLKREQ# signals must be connected.
- It is permitted for the Upstream Port to de-assert CLKREQ# when the Link is in the PCI-PM L1 or ASPM L1 states, or when the Link is in the L2/L3 Ready pseudo-state; CLKREQ# must be asserted by the Upstream Port when the Link is in any other state.
- All other specifications related to the CLKREQ# signal that are not specifically defined or modified by L1 PM Substates continue to apply.

If these requirements cannot be satisfied in a particular system, then L1 PM Substates must not be enabled.

Formatted: Font color: Black

IMPLEMENTATION NOTE

CLKREQ# Connection Topologies

For an Upstream component the connection topologies for the CLKREQ# signal can vary. A few examples of CLKREQ# connection topologies are described below. For the Downstream component these cases are essentially the same, however from the Upstream component's perspective, there are some key differences that are described below.

Example 1: Single Downstream Port with a single PLL connected to a single Upstream Port (see Figure 5-10).

In this platform configuration the Upstream component (A) has only a single CLKREQ# signal. The Upstream and Downstream Ports' CLKREQ# (A and B) signals are connected to each other. In this case, Upstream component (A), must assert CLKREQ# signal whenever it requires a reference clock.

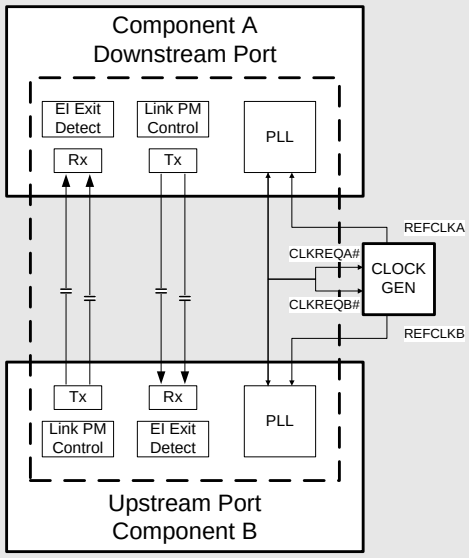


Figure 5-10: Downstream Port with a Single PLL

Example 2: Upstream component with multiple Downstream Ports, with a common shared PLL, connected to separate Downstream components (see Figure 5-11).

In this example configuration, there are three instances of CLKREQ# signal for the Upstream component (A), one per Downstream Port and a common shared CLKREQ# signal for the Upstream component (A). In this topology the Downstream Port CLKREQ# (CLKREQB#, CLKREQC#) signals are used to connect to the CLKREQ# signal of the Upstream Port of the Downstream components (B and C). The common shared CLKREQ# (CLKREQA#) signal for

the Upstream component is used to request the reference clock for the shared PLL. The PLL control logic in Upstream component (A) can only be turned off and CLKREQA# be de-asserted when both the Downstream Ports are in L1.1 or L1.2 Substates, and all internal (A) consumers of the PLL don't require a clock.

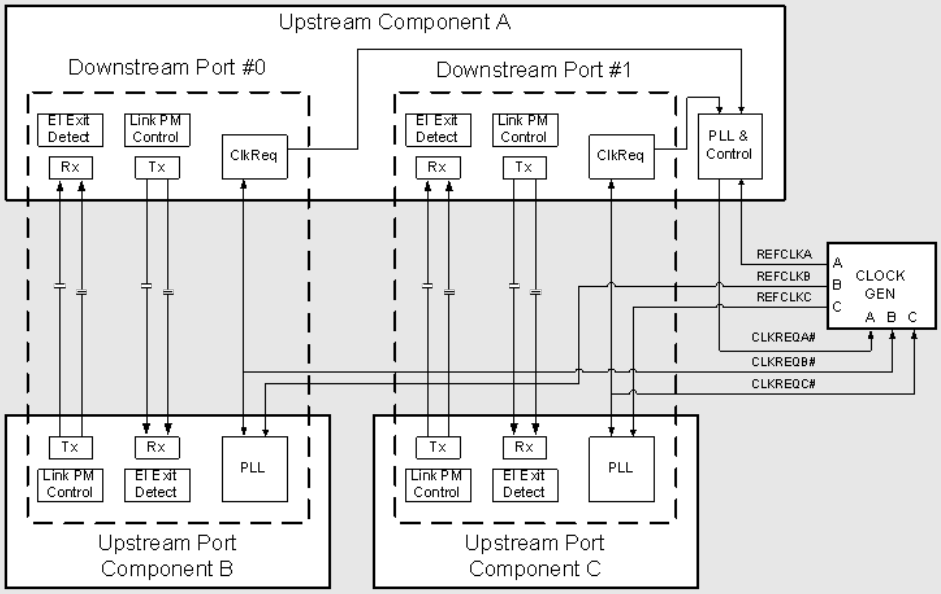


Figure 5-11: Multiple Downstream Ports with a shared PLL

It is necessary for board implementers to consider what CLKREQ# topologies will be supported by components in order to make appropriate board level connections to support L1 PM Substates and for the reference clock generation.

IMPLEMENTATION NOTE

Avoiding Unintended Interactions Between L1 PM Substates and the LTSSM

It is often the case that implementation techniques which save power will also increase the latency to return to normal operation. When implementing L1 PM Substates, it is important for the implementer to ensure that any added delays will not negatively interact with other elements of the platform. It is particularly important to ensure that LTSSM timeout conditions are not unintentionally triggered. Although typical implementations will not approach the latencies that would cause such interactions, the responsibility lies with the implementor to ensure that correct overall operation is achieved.

Formatted: Font color: Black

5.5.1. Entry conditions for L1 PM Substates and L1.0 Requirements

The Link is considered to be in PCI-PM L1.0 when the L1 PM Substate is L1.0 and the LTSSM entered L1 through PCI-PM compatible power management. The Link is considered to be in ASPM L1.0 when the L1 PM Substate is in L1.0 and LTSSM entered L1 through ASPM.

The following rules define how the L1.1 and L1.2 substates are entered:

- Both the Upstream and Downstream Ports must monitor the logical state of the CLKREQ# signal.
- When in PCI-PM L1.0 and the PCI-PM L1.2 Enable bit is Set, the L1.2 substate must be entered when CLKREQ# is de-asserted.
- When in PCI-PM L1.0 and the PCI-PM L1.1 Enable bit is Set, the L1.1 substate must be entered when CLKREQ# is de-asserted and the PCI-PM L1.2 Enable bit is Clear.
- When in ASPM L1.0 and the ASPM L1.2 Enable bit is Set, the L1.2 substate must be entered when CLKREQ# is de-asserted and all of the following conditions are true:
 - the reported snooped LTR value is greater than or equal to the value set by the LTR_L1.2_THRESHOLD Value and Scale fields, or there is no snoop service latency requirement
 - the reported non-snooped LTR value is greater than or equal to the value set by the LTR_L1.2_THRESHOLD Value and Scale fields, or there is no non-snoop service latency requirement
- When in ASPM L1.0 and the ASPM L1.1 Enable bit is Set, the L1.1 substate must be entered when CLKREQ# is de-asserted and the conditions for entering the L1.2 substate are not satisfied.

When the entry conditions for L1.2 are satisfied, the following rules apply:

- Both the Upstream and Downstream Ports must monitor the logical state of the CLKREQ# input signal.
- An Upstream Port must not de-assert CLKREQ# until the Link has entered L1.0.
- It is permitted for either Port to assert CLKREQ# to prevent the Link from entering L1.2.
- A Downstream Port intending to block entry into L1.2 must assert CLKREQ# before the Link enters L1.
- When CLKREQ# is de-asserted the Ports enter the L1.2.Entry substate of L1.2.

If a Downstream Port is in PCI-PM L1.0 and PCI-PM L1.1 Enable and/or PCI-PM L1.2 Enable are Set, or if a Downstream Port is in ASPM L1.0 and ASPM L1.1 Enable and/or ASPM L1.2 Enable are Set, and the Downstream Port initiates an exit to Recovery without having entered L1.1 or L1.2, the Downstream Port must assert CLKREQ# until the Link exits Recovery.

Comment [VS179]: Errata A23, CLKREQ# Rules

5.5.2. L1.1 Requirements

Both Upstream and Downstream Ports are permitted to deactivate mechanisms for electrical idle (EI) exit detection and Refclk activity detection if implemented, however both ports must maintain common mode.

5.5.2.1. Exit from L1.1

If either the Upstream or Downstream Port needs to initiate exit from L1.1, it must assert CLKREQ# until the Link exits Recovery. The Upstream Port must assert CLKREQ# on entry to Recovery, and must continue to assert CLKREQ# until the next entry into L1, or other state allowing CLKREQ# de-assertion.

- Next state is L1.0 if CLKREQ# is asserted.
 - The Refclk will eventually be turned on as defined in the PCI Express Mini CEM spec, which may be delayed according to the LTR advertized by the Upstream Port.

Figure 5-12 illustrates entry into L1.1 with exit driven by the Upstream Port.

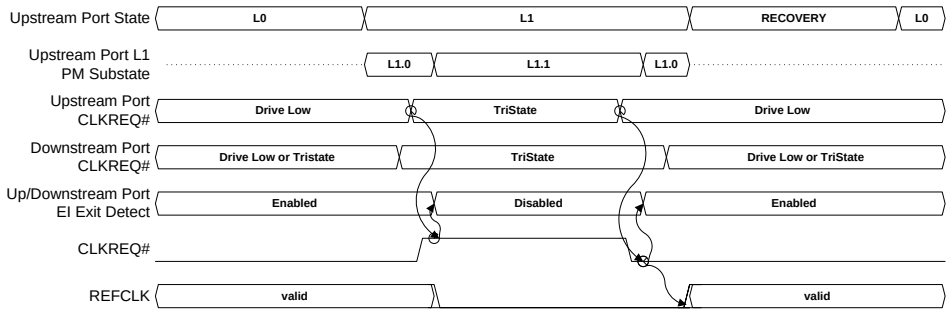


Figure 5-12: Example: L1.1 Waveforms Illustrating Upstream Port Initiated Exit

Figure 5-13 illustrates entry into L1.1 with exit driven by the Downstream Port.

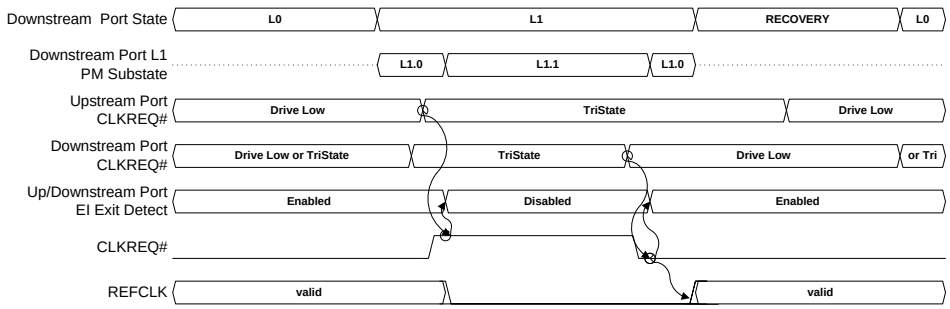


Figure 5-13: Example: L1.1 Waveforms Illustrating Downstream Port Initiated Exit

5.5.3. L1.2 Requirements

All Link and PHY state must be maintained during L1.2, or must be restored upon exit using implementation-specific means, and the LTSSM and corresponding Port state upon exit from L1.2 must be indistinguishable from the L1.0 LTSSM and Port state.

L1.2 has additional requirements that do not apply to L1.1. These requirements are documented in this section.

L1.2 has three substates, which are defined below (see Figure 5-14).

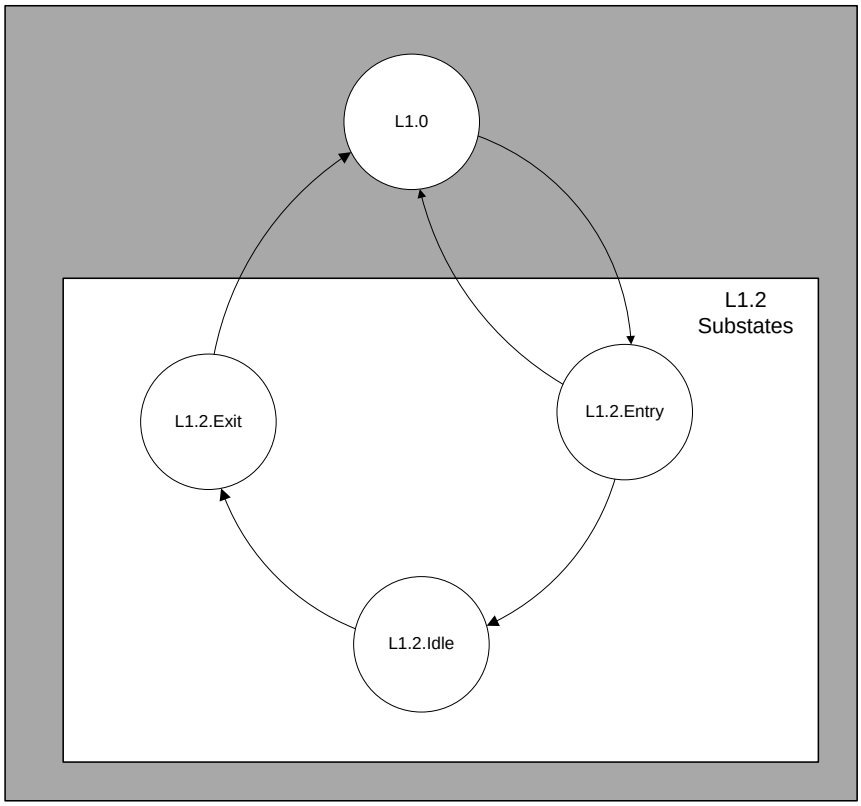


Figure 5-14: L1.2 Substates

5.5.3.1. L1.2.Entry

L1.2.Entry is a transitional state on entry into L1.2 to allow time for Refclk to turn off and to ensure both Ports have observed CLKREQ# de-asserted. The following rules apply to L1.2.Entry:

- Both Upstream and Downstream Ports continue to maintain common mode.
- Both Upstream and Downstream Ports may turn off their electrical idle (EI) exit detect circuitry.
- The Upstream and Downstream Ports must not assert CLKREQ# in this state.
- Refclk must be turned off within $T_{L1O_REFCLK_OFF}$.
- Next state is L1.0 if CLKREQ# is asserted, else the next state is L1.2.Idle after waiting for T_{POWER_OFF}.

Note that there is a boundary condition which can occur when one Port asserts CLKREQ# shortly after the other Port de-asserts CLKREQ#, but before the first Port has observed CLKREQ# de-

asserted. This is an unavoidable boundary condition that implementations must handle correctly. An example of this condition is illustrated in Figure 5-15.

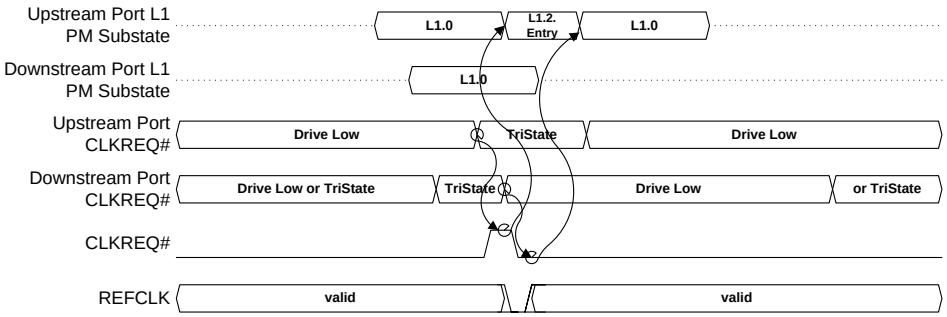


Figure 5-15: Example: Illustration of Boundary Condition due to Different Sampling of CLKREQ#

5.5.3.2. L1.2.Idle

When requirements for the entry into L1.2.Idle state (see Section 5.5.1) have been satisfied then the Ports enter the L1.2.Idle substate. The following rules apply in L1.2.Idle:

- Both Upstream and Downstream Ports may power-down any active logic, including circuits required to maintain common mode.
- The PHY of both Upstream and Downstream Ports may have their power removed.

The following rules apply for L1.2.Idle state when using the CLKREQ#-based mechanism:

- If either the Upstream or Downstream Port needs to exit L1.2, it must assert CLKREQ# after ensuring that $T_{L1.2}$ has been met.
- If the Downstream Port is initiating exit from L1, it must assert CLKREQ# until the Link exits Recovery. The Upstream Port must assert CLKREQ# on entry to Recovery, and must continue to assert CLKREQ# until the next entry into L1, or other state allowing CLKREQ# de-assertion.
- If the Upstream Port is initiating exit from L1, it must continue to assert CLKREQ# until the next entry into L1, or other state allowing CLKREQ# de-assertion.
- Both the Upstream and Downstream Ports must monitor the logical state of the CLKREQ# input signal.
- Next state is L1.2.Exit if CLKREQ# is asserted.

5.5.3.3. L1.2.Exit

This is a transitional state on exit from L1.2 to allow time for both devices to power up. In L1.2.Exit, the following rules apply:

- The PHYs of both Upstream and Downstream Ports must be powered.
- It must not be assumed that common mode has been maintained.

5.5.3.3.1. Exit from L1.2

- The following rules apply for L1.2 Exit using the CLKREQ#-based mechanism:
- Both Upstream and Downstream Ports must power up any circuits required for L1.0, including circuits required to maintain common mode.
- The Upstream and Downstream Ports must not change their driving state of CLKREQ# in this state.
- Refclk must be turned on no earlier than $T_{L1O_REFCLK_ON}$ minimum time, and may take up to the amount of time allowed according to the LTR advertized by the Endpoint before becoming valid.
- Next state is L1.0 after waiting for T_{POWER_ON}.
 - Common mode is permitted to be established passively during L1.0, and actively during Recovery. In order to ensure common mode has been established, the Downstream Port must maintain a timer, and the Downstream Port must continue to send TS1 training sequences until a minimum of $T_{COMMONMODE}$ has elapsed since the Downstream Port has started transmitting TS1 training sequences and has detected electrical idle exit on any Lane of the configured Link.

Comment [VS180]: Errata A21, ECN: L1 PM Substates with CLKREQ

Figure 5-16 illustrates the signal relationships and timing constraints associated with L1.2 entry and Upstream Port initiated exit.

Figure 5-17 illustrates the signal relationships and timing constraints associated with L1.2 entry and Downstream Port initiated exit.

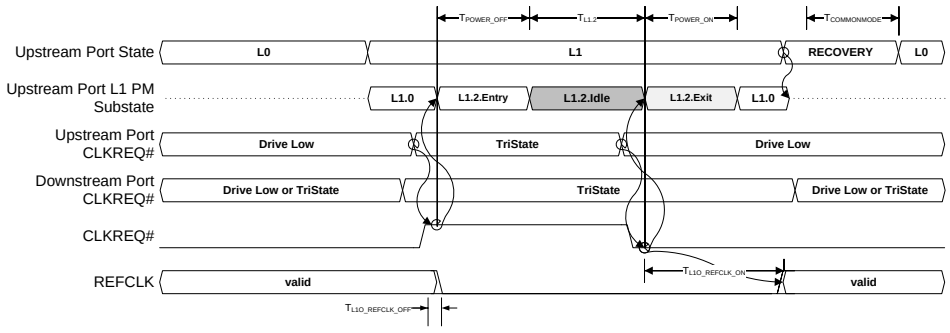


Figure 5-16: Example: L1.2 Waveforms Illustrating Upstream Port Initiated Exit

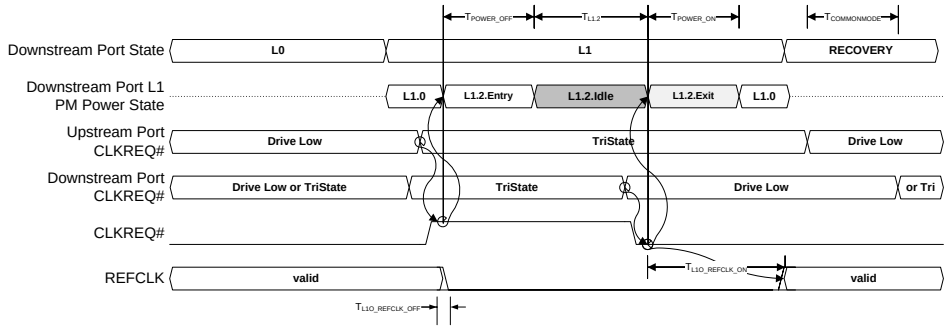


Figure 5-17: Example: L1.2 Waveforms Illustrating Downstream Port Initiated Exit

5.5.4. L1 PM Substates Configuration

L1 PM Substates is considered enabled on a Port when any combination of the ASPM L1.1 Enable, ASPM L1.2 Enable, PCI-PM L1.1 Enable and PCI-PM L1.2 Enable bits associated with that Port are Set.

An L1 PM Substate enable bit must only be Set in the Upstream and Downstream Ports on a Link when the corresponding supported capability bit is Set by both the Upstream and Downstream Ports on that Link, otherwise the behavior is undefined.

The Setting of any enable bit must be performed at the Downstream Port before the corresponding bit is permitted to be Set at the Upstream Port. If any L1 PM Substates enable bit is at a later time to be cleared, the enable bit(s) must be cleared in the Upstream Port before the corresponding enable bit(s) are permitted to be cleared in the Downstream Port.

If setting either or both of the enable bits for ASPM L1 PM Substates, both ports must be configured as described in this section while ASPM L1 is disabled.

If setting either or both of the enable bits for PCI-PM L1 PM Substates, both ports must be configured as described in this section while in D0.

Prior to setting either or both of the enable bits for L1.2, the values for T_{POWER_ON}, Common_Mode_Restore_Time, and, if the ASPM L1.2 Enable bit is to be Set, the LTR_L1.2_THRESHOLD (both Value and Scale fields) must be programmed.

The T_{POWER_ON} and Common_Mode_Restore_Time fields must be programmed to the appropriate values based on the components and AC coupling capacitors used in the connection linking the two components. The determination of these values is design implementation specific.

When both the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are cleared, it is not required to program the T_{POWER_ON}, Common_Mode_Restore_Time, and LTR_L1.2_THRESHOLD Value and Scale fields, and hardware must not rely on these fields to have any particular values.

When programming LTR_L1.2_THRESHOLD Value and Scale fields, identical values must be programmed in both Ports.

5.5.5. L1 PM Substates Timing Parameters

The following table defines the timing parameters associated with the L1.2 substates mechanism.

Table 5-11: Timing Parameters

Parameter	Description	Min	Max	Units
T_{POWER_OFF}	CLKREQ# de-assertion to entry into the L1.2.Idle substate		2	µs
$T_{COMMONMODE}$	Restoration of Refclk to restoration of common mode established through active transmission of TS1 training sequences (see Section 5.5.3.3.1)	Programmable in range from 0 to 255		µs
$T_{L1O_REFCLK_OFF}$	CLKREQ# de-assertion to Refclk reaching idle electrical state when entering L1.2	0	100	ns
$T_{L1O_REFCLK_ON}$	CLKREQ# assertion to Refclk valid when exiting L1.2	T_{POWER_ON}	LTR value advertized by the Endpoint	µs
T_{POWER_ON}	The minimum amount of time that each component must wait in L1.2.Exit after sampling CLKREQ# asserted before actively driving the interface to ensure no device is ever actively driving into an unpowered component.	Set in the L1 PM Substates Control 2 Register (range from 0 to 3100)		µs
$T_{L1.2}$	Time a Port must stay in L1.2 when CLKREQ# must remain inactive	4		µs

~~5.5.~~5.6. Auxiliary Power Support

~~5.5.1.~~5.6.1. Auxiliary Power Enabling

The PCI-PM specification requires that a Function must support PME generation in order to consume the maximum allowance of auxiliary current (375 mA vs. 20 mA). However, there are instances where Functions need to consume power even if they are "PME Disabled," or PME incapable by design. One example is a component with its system management mode active during a system low power state.

PCI Express PM provides a new Aux Power PM Enable bit in the Device Control register that provides the means for enabling a Function to draw the maximum allowance of auxiliary current independent of its level of support for PME generation.

A Function requests Aux power allocation by specifying a non-zero value in the Aux_Current field of the PMC register. Refer to Chapter 7 for the Aux Power PM Enable register bit assignment, and access mechanism.

Legacy PCI-PM software is unaware of this new bit and will only be able to enable Aux current to a given Function based on the Function's reported PME_Support field, the Aux_Current field value, and the Function's PME_En bit.

Allocation of Aux power using Aux Power PM Enable is determined as follows:

Aux Power PM Enable = 1b:

Aux power is allocated as requested in the Aux_Current field of the PMC register, independent of the PME_En bit in the PMSCR. The PME_En bit still controls the ability to master PME.

Aux Power PM Enable = 0b:

Aux power allocation is controlled by the PME_En bit as defined in the *PCI Bus Power Management Interface Specification*.

The Aux Power PM Enable bit is sticky (see Section 7.4) so its state is preserved in the $D3_{cold}$ state, and is not affected by the transitions from the $D3_{cold}$ state to the $D0_{uninitialized}$ state.

Typically, Aux power is required to support the Beacon wakeup mechanism, and components supporting Beacon must not consume Aux power unless enabled by software to do so. To enable finer granularity of power consumption, Root Ports and Switch Ports should support independent control of Aux power consumption for each Port. However, it is permitted to logically OR the Aux power enables for multiple Ports and to combine the Aux power control for those Ports so long as the combined Aux power consumption does not exceed the sum of the amounts enabled by software.

~~5.6.~~5.7. Power Management System Messages and DLLPs

Table 5-12 defines the location of each PM packet in the PCI Express stack.

Table 5-12: Power Management System Messages and DLLPs

Packet	Type
PM_Enter_L1	DLLP
PM_Enter_L23	DLLP
PM_Active_State_Request_L1	DLLP
PM_Request_Ack	DLLP
PM_Active_State_Nak	Transaction Layer Message
PM_PME	Transaction Layer Message
PME_Turn_Off	Transaction Layer Message
PME_TO_Ack	Transaction Layer Message

For information on the structure of the power management DLLPs, refer to Section 3.4.

Power Management Messages follow the general rules for all Messages. Power Management Message fields follow the following rules:

- Length field is Reserved.
- Attribute field must be set to the default values (all 0's).
- Address field is Reserved.
- Requester ID – see Table 2-21 in Section 2.2.8.2.
- Traffic Class field must use the default class (TC0).

6

6. System Architecture

This chapter addresses various aspects of PCI Express interconnect architecture in a platform context.

6.1. Interrupt and PME Support

The PCI Express interrupt model supports two mechanisms:

- INTx emulation
- Message Signaled Interrupt (MSI/MSI-X)

For legacy compatibility, PCI Express provides a PCI INTx emulation mechanism to signal interrupts to the system interrupt controller (typically part of the Root Complex). This mechanism is compatible with existing PCI software, and provides the same level and type of service as the corresponding PCI interrupt signaling mechanism and is independent of system interrupt controller specifics. This legacy compatibility mechanism allows boot device support without requiring complex BIOS-level interrupt configuration/control service stacks. It virtualizes PCI physical interrupt signals by using an in-band signaling mechanism.

In addition to PCI INTx compatible interrupt emulation, PCI Express requires support of MSI or MSI-X or both. The PCI Express MSI and MSI-X mechanisms are compatible with those defined in the *PCI Local Bus Specification*.

6.1.1. Rationale for PCI Express Interrupt Model

PCI Express takes an evolutionary approach from PCI with respect to interrupt support.

As required for PCI/PCI-X interrupt mechanisms, each device Function is required to differentiate between INTx (legacy) and MSI (native) modes of operation. The device complexity required to support both schemes is no different than that for PCI/PCI-X devices. The advantages of this approach include:

- Compatibility with existing PCI Software Models
- Direct support for boot devices
- Easier End of Life (EOL) for INTx legacy mechanisms.

The existing software model is used to differentiate legacy (INTx) vs. MSI modes of operation; thus, no special software support is required for PCI Express.

6.1.2. PCI Compatible INTx Emulation

PCI Express supports the PCI interrupts as defined in the P*CI Local Bus Specification* including the Interrupt Pin and Interrupt Line registers of the PCI Configuration Space for PCI device Functions. PCI Express devices support these registers for backwards compatibility; actual interrupt signaling uses in-band Messages rather than being signaled using physical pins.

Two types of Messages are defined, Assert_INTx and Deassert_INTx, for emulation of PCI INTx signaling, where x is A, B, C, and D for respective PCI interrupt signals. These Messages are used to provide "virtual wires" for signaling interrupts across a Link. Switches collect these virtual wires and present a combined set at the Switch's Upstream Port. Ultimately, the virtual wires are routed to the Root Complex which maps the virtual wires to system interrupt resources. Devices must use assert/de-assert Messages in pairs to emulate PCI interrupt level-triggered signaling. Actual mapping of PCI Express INTx emulation to system interrupts is implementation specific as is mapping of physical interrupt signals in conventional PCI.

The legacy INTx emulation mechanism may be deprecated in a future version of this specification.

6.1.3. INTx Emulation Software Model

The software model for legacy INTx emulation matches that of PCI. The system BIOS reporting of chipset/platform interrupt mapping and the association of each device Function's interrupt with PCI interrupt lines is handled in exactly the same manner as with conventional PCI systems. Legacy software reads from each device Function's Interrupt Pin register to determine if the Function is interrupt driven. A value between 01h and 04h indicates that the Function uses an emulated interrupt pin to generate an interrupt.

Note that similarly to physical interrupt signals, the INTx emulation mechanism may potentially cause spurious interrupts that must be handled by the system software.

6.1.4. Message Signaled Interrupt (MSI/MSI-X) Support

MSI/MSI-X interrupt support, which is optional for PCI 3.0 devices, is required for PCI Express devices. All PCI Express device Functions that are capable of generating interrupts must support MSI or MSI-X or both. The MSI and MSI-X mechanisms deliver interrupts by performing memory write transactions. MSI and MSI-X are edge-triggered interrupt mechanisms; neither the *PCI Local Bus Specification* nor this specification support level-triggered MSI/MSI-X interrupts. Certain PCI devices and their drivers rely on INTx-type level-triggered interrupt behavior (addressed by the PCI Express legacy INTx emulation mechanism). To take advantage of the MSI or MSI-X capability and edge-triggered interrupt semantics, these devices and their drivers may have to be redesigned.



IMPLEMENTATION NOTE

Per Vector Masking with MSI/MSI-X

Devices and drivers that use MSI or MSI-X have the challenge of coordinating exactly when new interrupt messages are generated. If hardware fails to send an interrupt message that software expects, an interrupt event might be "lost." If hardware sends an interrupt message that software is not expecting, a "spurious" interrupt might result.

Per-Vector Masking (PVM) is an architected feature for MSI and MSI-X that can be used to assist in this coordination. For example, when a software interrupt service routine begins, it can mask the vector to help avoid "spurious" interrupts. After the interrupt service routine services all the interrupt conditions that it is aware of, it can unmask the vector. If any interrupt conditions remain, hardware is required to generate a new interrupt message, guaranteeing that no interrupt events are lost.

PVM is a standard feature with MSI-X and an optional feature for MSI. For devices that implement MSI, implementing PVM as well is highly recommended.

A Legacy Endpoint that implements MSI is required to support either the 32-bit or 64-bit Message Address version of the MSI Capability structure. A PCI Express Endpoint that implements MSI is required to support the 64-bit Message Address version of the MSI Capability structure.

The Requester of an MSI/MSI-X transaction must set the No Snoop and Relaxed Ordering attributes of the Transaction Descriptor to 0b. A Requester of an MSI/MSI-X transaction is permitted to set the ID-Based Ordering (IDO) attribute if use of the IDO attribute is enabled.

Note that, unlike INTx emulation Messages, MSI/MSI-X transactions are not restricted to the TC0 traffic class.



IMPLEMENTATION NOTE

Synchronization of Data Traffic and Message Signaled Interrupts

MSI/MSI-X transactions are permitted to use the TC that is most appropriate for the device's programming model. This is generally the same TC as is used to transfer data; for legacy I/O, TC0 should be used.

If a device uses more than one TC, it must explicitly ensure that proper synchronization is maintained between data traffic and interrupt Message(s) not using the same TC. Methods for ensuring this synchronization are implementation specific. One option is for a device to issue a zero-length Read (as described in Section 2.2.5) using each additional TC used for data traffic prior to issuing the MSI/MSI-X transaction. Other methods are also possible. Note, however, that platform software (e.g., a device driver) is generally only capable of issuing transactions using TC0.

6.1.5. PME Support

PCI Express supports power management events from native PCI Express devices as well as PME-capable PCI devices.

PME signaling is accomplished using an in-band Transaction Layer PME Message (PM_PME) as described in Chapter 5.

6.1.6. Native PME Software Model

PCI Express-aware software can enable a mode where the Root Complex signals PME via an interrupt. When configured for native PME support, a Root Port receives the PME Message and sets the PME Status bit in its Root Status register. If software has set the PME Interrupt Enable bit in the Root Control register to 1b, the Root Port then generates an interrupt.

If the Root Port is enabled for level-triggered interrupt signaling using the INTx messages, the virtual INTx wire must be asserted whenever and as long as all of the following conditions are satisfied:

- The Interrupt Disable bit in the Command register is set to 0b.
- The PME Interrupt Enable bit in the Root Control register is set to 1b.
- The PME Status bit in the Root Status register is set.

Note that all other interrupt sources within the same Function will assert the same virtual INTx wire when requesting service.

If the Root Port is enabled for edge-triggered interrupt signaling using MSI or MSI-X, an interrupt message must be sent every time the logical AND of the following conditions transitions from FALSE to TRUE:

- The associated vector is unmasked (not applicable if MSI does not support PVM).
- The PME Interrupt Enable bit in the Root Control register is set to 1b.
- The PME Status bit in the Root Status register is set.

Note that PME and Hot-Plug Event interrupts (when both are implemented) always share the same MSI or MSI-X vector, as indicated by the Interrupt Message Number field in the PCI Express Capabilities register.

The software handler for this interrupt can determine which device sent the PME Message by reading the PME Requester ID field in the Root Status register in a Root Port. It dismisses the interrupt by writing a 1b to the PME Status bit in the Root Status register. Refer to Section 7.8.14 for more details.

Root Complex Event Collectors provide support for the above described functionality for Root Complex Integrated Endpoints.

6.1.7. Legacy PME Software Model

Legacy software, however, will not understand this mechanism for signaling PME. In the presence of legacy system software, the system power management logic in the Root Complex receives the PME Message and informs system software through an implementation specific mechanism. The Root Complex may utilize the Requester ID in the PM_PME to inform system software which device caused the power management event.

Since it is delivered by a Message, PME has edge-triggered semantics in PCI Express, which differs from the level-triggered PME mechanism used for conventional PCI. It is the responsibility of the Root Complex to abstract this difference from system software to maintain compatibility with conventional PCI systems.

6.1.8. Operating System Power Management Notification

In order to maintain compatibility with non-PCI Express-aware system software, system power management logic must be configured by firmware to use the legacy mechanism of signaling PME by default. PCI Express-aware system software must notify the firmware prior to enabling native, interrupt-based PME signaling. In response to this notification, system firmware must, if needed, reconfigure the Root Complex to disable legacy mechanisms of signaling PME. The details of this firmware notification are beyond the scope of this specification, but since it will be executed at system run-time, the response to this notification must not interfere with system software. Therefore, following control handoff to the operating system, firmware must not write to available system memory or any PCI Express resources (e.g., Configuration Space structures) owned by the operating system.

6.1.9. PME Routing Between PCI Express and PCI Hierarchies

PME-capable conventional PCI and PCI-X devices assert the PME# pin to signal a power management event. The PME# signal from PCI or PCI-X devices may either be converted to a PCI Express in-band PME Message by a PCI Express-PCI Bridge or routed directly to the Root Complex.

If the PME# signal from a PCI or PCI-X device is routed directly to the Root Complex, it signals system software using the same mechanism used in present PCI systems. A Root Complex may optionally provide support for signaling PME from PCI or PCI-X devices to system software via an interrupt. In this scenario, it is recommended for the Root Complex to detect the Bus, Device and Function Number of the PCI or PCI-X device that asserted PME#, and use this information to fill in the PME Requester ID field in the Root Port that originated the hierarchy containing the PCI or PCI-X device. If this is not possible, the Root Complex may optionally write the Requester ID of the Root Port to this field.

Since Root Complex Integrated Endpoints are not contained in any of the hierarchy domains originated by Root Ports, Root Complex Integrated Endpoints not associated with a Root Complex Event Collector signal system software of a PME using the same mechanism used in present PCI systems. A Root Complex Event Collector, if implemented, enables the PCI Express Native PME model for associated Root Complex Integrated Endpoints.

6.2. Error Signaling and Logging

In this document, errors which must be checked and errors which may optionally be checked are identified. Each such error is associated either with the Port or with a specific device (or Function in a multi-Function device), and this association is given along with the description of the error. This section will discuss how errors are classified and reported.

6.2.1. Scope

This section explains the error signaling and logging requirements for PCI Express components. This includes errors which occur on the PCI Express interface itself, those errors which occur on behalf of transactions initiated on PCI Express, and errors which occur within a component and are related to the PCI Express interface. This section does not focus on errors which occur within the component that are unrelated to a PCI Express interface. This type of error signaling is better handled through proprietary methods employing device-specific interrupts.

PCI Express defines two error reporting paradigms: the baseline capability and the Advanced Error Reporting Capability. The baseline error reporting capabilities are required of all PCI Express devices and define the minimum error reporting requirements. The Advanced Error Reporting Capability is defined for more robust error reporting and is implemented with a specific PCI Express Capability structure (refer to Chapter 7 for a definition of this optional capability). This section explicitly calls out all error handling differences between the baseline and the Advanced Error Reporting Capability.

All PCI Express devices support existing, non-PCI Express-aware, software for error handling by mapping PCI Express errors to existing PCI reporting mechanisms, in addition to the PCI Express-specific mechanisms.

6.2.2. Error Classification

PCI Express errors can be classified as two types: Uncorrectable errors and Correctable errors. This classification separates those errors resulting in functional failure from those errors resulting in degraded performance. Uncorrectable errors can further be classified as Fatal or Non-Fatal (see Figure 6-1).

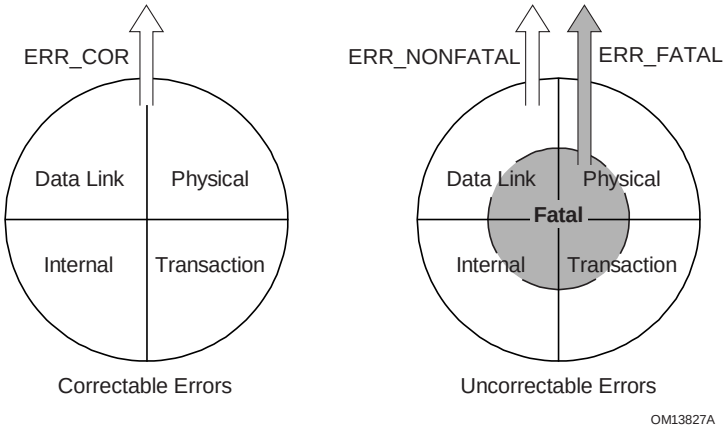


Figure 6-1: Error Classification

Classification of error severity as Fatal, Uncorrectable, and Correctable provides the platform with mechanisms for mapping the error to a suitable handling mechanism. For example, the platform might choose to respond to correctable errors with low priority, performance monitoring software.

Such software could count the frequency of correctable errors and provide Link integrity information. On the other hand, a platform designer might choose to map Fatal errors to a system-wide reset. It is the decision of the platform designer to map these PCI Express severity levels onto platform level severities.

6.2.2.1. Correctable Errors

Correctable errors include those error conditions where hardware can recover without any loss of information. Hardware corrects these errors and software intervention is not required. For example, an LCRC error in a TLP that might be corrected by Data Link Level Retry is considered a correctable error. Measuring the frequency of Link-level correctable errors may be helpful for profiling the integrity of a Link.

Correctable errors also include transaction-level cases where one agent detects an error with a TLP, but another agent is responsible for taking any recovery action if needed, such as re-attempting the operation with a separate subsequent transaction. The detecting agent can be configured to report the error as being correctable since the recovery agent may be able to correct it. If recovery action is indeed needed, the recovery agent must report the error as uncorrectable if the recovery agent decides not to attempt recovery.

The triggering of Downstream Port Containment (DPC) is not handled as an error, but it can be signaled as if it were a correctable error, since software that takes advantage of DPC can sometimes recover from the uncorrectable error that triggered DPC. See Section 6.2.10.

Comment [VS181]: ECN: Downstream Port containment (DPC)

6.2.2.2. Uncorrectable Errors

Uncorrectable errors are those error conditions that impact functionality of the interface. There is no mechanism defined in this specification to correct these errors. Reporting an uncorrectable error is analogous to asserting SERR# in PCI/PCI-X. For more robust error handling by the system, this specification further classifies uncorrectable errors as Fatal and Non-fatal.

6.2.2.2.1. Fatal Errors

Fatal errors are uncorrectable error conditions which render the particular Link and related hardware unreliable. For Fatal errors, a reset of the components on the Link may be required to return to reliable operation. Platform handling of Fatal errors, and any efforts to limit the effects of these errors, is platform implementation specific.

6.2.2.2.2. Non-Fatal Errors

Non-fatal errors are uncorrectable errors which cause a particular transaction to be unreliable but the Link is otherwise fully functional. Isolating Non-fatal from Fatal errors provides Requester/Receiver logic in a device or system management software the opportunity to recover from the error without resetting the components on the Link and disturbing other transactions in progress. Devices not associated with the transaction in error are not impacted by the error.

6.2.3. Error Signaling

There are three complementary mechanisms which allow the agent detecting an error to alert the system or another device that an error has occurred. The first mechanism is through a Completion Status, the second method is with in-band error Messages, and the third is with Error Forwarding (also known as data poisoning).

Note that it is the responsibility of the agent detecting the error to signal the error appropriately.

Section 6.2.7 describes all the errors and how the hardware is required to respond when the error is detected.

6.2.3.1. Completion Status

The Completion Status field (when status is not Successful Completion) in the Completion header indicates that the associated Request failed (see Section 2.2.8.10). This is one method of error reporting which enables the Requester to associate an error with a specific Request. In other words, since Non-Posted Requests are not considered complete until after the Completion returns, the Completion Status field gives the Requester an opportunity to "fix" the problem at some higher level protocol (outside the scope of this specification). For example, if a Read is issued to prefetchable Memory Space and the Completion returns with an Unsupported Request Completion Status, the Requester would not be in violation of this specification if it chose to reissue the Read Request. Note that from a PCI Express point of view, the reissued Read Request is a distinct Request, and there is no relationship (on PCI Express) between the initial Request and the reissued Request.

6.2.3.2. Error Messages

Error Messages are sent to the Root Complex for reporting the detection of errors according to the severity of the error.

Error messages that originate from PCI Express or Legacy Endpoints are sent to corresponding Root Ports. Errors that originate from a Root Port itself are reported through the same Root Port.

If a Root Complex Event Collector is implemented, errors that originate from a Root Complex Integrated Endpoint may optionally be sent to the corresponding Root Complex Event Collector. Errors that originate from a Root Complex Integrated Endpoint are reported in a Root Complex Event Collector residing on the same Logical Bus as the Root Complex Integrated Endpoint. The Root Complex Event Collector must explicitly declare supported Root Complex Integrated Endpoints as part of its capabilities; each Root Complex Integrated Endpoint must be associated with no more than one Root Complex Event Collector.

When multiple errors of the same severity are detected, the corresponding error Messages with the same Requester ID may be merged for different errors of the same severity. At least one error Message must be sent for detected errors of each severity level. Note, however, that the detection of a given error in some cases will preclude the reporting of certain errors. Refer to Section 6.2.3.2.3. Also note special rules in Section 6.2.4 regarding non-Function-specific errors in multi-Function devices.

Table 6-1: Error Messages

Error Message	Description
ERR_COR	This Message is issued when the Function or Device detects a correctable error on the PCI Express interface. Refer to Section 6.2.2.1 for the definition of a correctable error.
ERR_NONFATAL	This Message is issued when the Function or Device detects a Non-fatal, uncorrectable error on the PCI Express interface. Refer to Section 6.2.2.2.2 for the definition of a Non-fatal, uncorrectable error.
ERR_FATAL	This Message is issued when the Function or Device detects a Fatal, uncorrectable error on the PCI Express interface. Refer to Section 6.2.2.2.1 for the definition of a Fatal, uncorrectable error.

For these Messages, the Root Complex identifies the initiator of the Message by the Requester ID of the Message header. The Root Complex translates these error Messages into platform level events.



IMPLEMENTATION NOTE

Use of ERR_COR, ERR_NONFATAL, and ERR_FATAL

In the 1.0 and 1.0a Base specifications, a given error was either correctable, non-fatal, or fatal. Assuming signaling was enabled, correctable errors were always signaled with ERR_COR, non-fatal errors were always signaled with ERR_NONFATAL, and fatal errors were always signaled with ERR_FATAL.

In subsequent specifications that support Role-Based Error Reporting, non-fatal errors are sometimes signaled with ERR_NONFATAL, sometimes signaled with ERR_COR, and sometimes not signaled at all, depending upon the role of the agent that detects the error and whether the agent implements AER (see Section 6.2.3.2.4). On some platforms, sending ERR_NONFATAL will preclude another agent from attempting recovery or determining the ultimate disposition of the error. For cases where the detecting agent is not the appropriate agent to determine the ultimate disposition of the error, a detecting agent with AER can signal the non-fatal error with ERR_COR, which serves as an advisory notification to software. For cases where the detecting agent is the appropriate one, the agent signals the non-fatal error with ERR_NONFATAL.

For a given uncorrectable error that's normally non-fatal, if software wishes to avoid continued hierarchy operation upon the detection of that error, software can configure detecting agents that implement AER to escalate the severity of that error to fatal. A detecting agent (if enabled) will always signal a fatal error with ERR_FATAL, regardless of the agent's role.

Software should recognize that a single transaction can be signaled by multiple agents using different types of error Messages. For example, a poisoned TLP might be signaled by intermediate Receivers with ERR_COR, while the ultimate destination Receiver might signal it with ERR_NONFATAL.

6.2.3.2.1. Uncorrectable Error Severity Programming (Advanced Error Reporting)

For device Functions implementing the Advanced Error Reporting Capability, the Uncorrectable Error Severity register allows each uncorrectable error to be programmed to Fatal or Non-Fatal. Uncorrectable errors are not recoverable using defined PCI Express mechanisms. However, some platforms or devices might consider a particular error fatal to a Link or device while another platform considers that error non-fatal. The default value of the Uncorrectable Error Severity register serves as a starting point for this specification but the register can be reprogrammed if the device driver or platform software requires more robust error handling.

Baseline error handling does not support severity programming.

6.2.3.2.2. Masking Individual Errors

Section 6.2.7 lists all the errors governed by this specification and describes when each of the above error Messages are issued. The transmission of these error Messages by class (correctable, non-fatal, fatal) is enabled using the Reporting Enable bits of the Device Control register (see Section 7.8.4) or the SERR# Enable bit in the PCI Command register (see Section 7.5.1.1).

For devices implementing the Advanced Error Reporting Capability the Uncorrectable Error Mask register and Correctable Error Mask register allows each error condition to be masked independently. If Messages for a particular class of error are not enabled by the combined settings in the Device Control register and the PCI Command register, then no Messages of that class will be sent regardless of the values for the corresponding mask register.

If an individual error is masked when it is detected, its error status bit is still affected, but no error reporting Message is sent to the Root Complex, and the error is not recorded in the Header Log, TLP Prefix Log, or ~~and~~ First Error Pointer ~~registers are unmodified~~.

Comment [VS182]: Errata B15

6.2.3.2.3. Error Pollution

Error pollution can occur if error conditions for a given transaction are not isolated to the most significant occurrence. For example, assume the Physical Layer detects a Receiver Error. This error is detected at the Physical Layer and an error is reported to the Root Complex. To avoid having this error propagate and cause subsequent errors at upper layers (for example, a TLP error at the Data Link Layer), making it more difficult to determine the root cause of the error, subsequent errors which occur for the same packet will not be reported by the Data Link or Transaction layers. Similarly, when the Data Link Layer detects an error, subsequent errors which occur for the same packet will not be reported by the Transaction Layer. This behavior applies only to errors that are associated with a particular packet – other errors are reported for each occurrence.

Corrected Internal Errors are errors whose effect has been masked or worked around by a component; refer to Section 6.2.9 for details. Therefore, Corrected Internal Errors do not contribute to error pollution and should be reported when detected.

For errors detected in the Transaction layer and Uncorrectable Internal Errors, it is permitted and recommended that no more than one error be reported for a single received TLP, and that the following precedence (from highest to lowest) be used:

- Uncorrectable Internal Error
- Receiver Overflow
- Flow Control Protocol Error
- ~~ECRC Check Failed~~
- Malformed TLP
- ECRC Check Failed
- AtomicOp Egress Blocked
- TLP Prefix Blocked
- ACS Violation

Comment [VS183]: Moved per Errata A24

- MC Blocked TLP
- Unsupported Request (UR), Completer Abort (CA), or Unexpected Completion[68]
- Poisoned TLP Received or Poisoned TLP Egress Blocked

The Completion Timeout error is not in the above precedence list, since it is not detected by processing a received TLP. Errors listed under the same bullet are mutually exclusive, so their relative order does not matter.

~~Here's an example of the rationale behind the precedence list. If an ECRC Check fails for a given TLP, the entire contents of the TLP including its header is potentially corrupt, so it makes little sense to report errors like Malformed TLP or Unsupported Request detected with the TLP.~~

Comment [VS184]: ECN: Enhanced DPC

Comment [VS185]: ECN: Enhanced DPC

Comment [VS186]: Errata A24 (resolution 10/13)

6.2.3.2.4. Advisory Non-Fatal Error Cases

In some cases the detector of a non-fatal error is not the most appropriate agent to determine whether the error is recoverable or not, or if it even needs any recovery action at all. For example, if software attempts to perform a configuration read from a non-existent device or Function, the resulting UR Status in the Completion will signal the error to software, and software does not need for the Completer in addition to signal the error by sending an ERR_NONFATAL Message. In fact, on some platforms, signaling the error with ERR_NONFATAL results in a System Error, which breaks normal software probing.

"Advisory Non-Fatal Error" cases are predominantly determined by the role of the detecting agent (Requester, Completer, or Receiver) and the specific error. In such cases, an agent with AER signals the non-fatal error (if enabled) by sending an ERR_COR Message as an advisory to software, instead of sending ERR_NONFATAL. An agent without AER sends no error Message for these cases, since software receiving ERR_COR would be unable to distinguish Advisory Non-Fatal Error cases from the correctable error cases used to assess Link integrity.

Following are the specific cases of Advisory Non-Fatal Errors. Note that multiple errors from the same or different error classes (correctable, non-fatal, fatal) may be present with a single TLP. For example, an unexpected Completion might also be poisoned. Refer to Section 6.2.3.2.3 for requirements and recommendations on reporting multiple errors. For the previous example, it is recommended that Unexpected Completion be reported, and that Poisoned TLP Received not be reported.

If software wishes for an agent with AER to handle what would normally be an Advisory Non-Fatal Error case as being more serious, software can escalate the severity of the uncorrectable error to fatal, in which case the agent (if enabled) will signal the error with ERR_FATAL.

This section covers Advisory Non-Fatal Error handling for errors managed by the PCI Express Extended Capability and AER. Section 6.2.10.3 covers the RP PIO error handling mechanism for Root Ports that support RP Extensions for DPC. RP PIO Advisory Non-Fatal Errors are similar in concept to AER Advisory Non-Fatal Errors, but apply to different error cases and are managed by different controls.

Comment [VS187]: ECN: Enhanced DPC

[68] ~~These are mutually exclusive errors, so their relative order does not matter.~~

6.2.3.2.4.1. Completer Sending a Completion with UR/CA Status

A Completer generally sends a Completion with an Unsupported Request or Completer Abort (UR/CA) Status to signal an uncorrectable error for a Non-Posted Request.[69] If the severity of the UR/CA error[70] is non-fatal, the Completer must handle this case as an Advisory Non-Fatal Error.[71] A Completer with AER signals the non-fatal error (if enabled) by sending an ERR_COR Message. A Completer without AER sends no error Message for this case.

Even though there was an uncorrectable error for this specific transaction, the Completer must handle this case as an Advisory Non-Fatal Error, since the Requester upon receiving the Completion with UR/CA Status is responsible for reporting the error (if necessary) using a Requester-specific mechanism (see Section 6.2.3.2.5).

6.2.3.2.4.2. Intermediate Receiver

When a Receiver that's not serving as the ultimate PCI Express destination for a TLP detects[72] a non-fatal error with the TLP, this "intermediate" Receiver must handle this case as an Advisory Non-Fatal Error.[73] A Receiver with AER signals the error (if enabled) by sending an ERR_COR Message. A Receiver without AER sends no error Message for this case. An exception to the intermediate Receiver case for Root Complexes (RCs) is noted below.

An example where the intermediate Receiver case occurs is a Switch that detects poison or bad ECRC in a TLP that it is routing. Even though this was an uncorrectable (but non-fatal) error at this point in the TLP's route, the intermediate Receiver handles it as an Advisory Non-Fatal Error, so that the ultimate Receiver of the TLP (i.e., the Completer for a Request TLP, or the Requester for a Completion TLP) is not precluded from handling the error more appropriately according to its error settings. For example, a given Completer that detects poison in a Memory Write Request[74] might have the error masked (and thus go unsignaled), whereas a different Completer in the same hierarchy might signal that error with ERR_NONFATAL.

A Poisoned TLP Egress Blocked error is never handled as an intermediate Receiver case since it is not detected as a part of processing a received TLP.

Comment [VS188]: ECN: Enhanced DPC

If an RC detects a non-fatal error with a TLP it normally would forward peer-to-peer between Root Ports, but the RC does not support propagating the error related information (e.g., a TLP Digest,

[69] If the Completer is returning data in a Completion, and the data is bad or suspect, the Completer is permitted to signal the error using the Error Forwarding (Data Poisoning) mechanism instead of handling it as a UR or CA.

[70] Certain other errors (e.g., ACS Violation) with a Non-Posted Request also result in the Completer sending a Completion with UR or CA Status. If the severity of the error (e.g., ACS Violation) is non-fatal, the Completer must also handle this case as an Advisory Non-Fatal Error. However, see Section 2.7.2.2 regarding certain Requests with Poisoned data that must be handled as uncorrectable errors.

[71] If the severity is fatal, the error is not an Advisory Non-Fatal Error, and must be signaled (if enabled) with ERR_FATAL.

[72] If the Receiver does not implement ECRC Checking or ECRC Checking is not enabled, the Receiver will not detect an ECRC Error.

[73] If the severity is fatal, the error is not an Advisory Non-Fatal Error, and must be signaled (if enabled) with ERR_FATAL.

[74] See Section 2.7.2.2 for special rules that apply for poisoned Memory Write Requests.

EP bit, or equivalent) with the forwarded transaction, the RC must signal the error (if enabled) with ERR_NONFATAL and also must not forward the transaction. An example is an RC needing to forward a poisoned TLP peer-to-peer between Root Ports, but the RC's internal fabric does not support poison indication.

6.2.3.2.4.3. Ultimate PCI Express Receiver of a Poisoned TLP

When a poisoned TLP is received by its ultimate PCI Express destination, if the severity is non-fatal and the Receiver deals with the poisoned data in a manner that permits continued operation, the Receiver must handle this case[75] as an Advisory Non-Fatal Error.[76] A Receiver with AER signals the error (if enabled) by sending an ERR_COR Message. A Receiver without AER sends no error Message for this case. Refer to Section 2.7.2.2 for special rules that apply for poisoned Memory Write Requests.

Comment [VS189]: Errata A12

An example is a Root Complex that receives a poisoned Memory Write TLP that targets host memory. If the Root Complex propagates the poisoned data along with its indication to host memory, it signals the error (if enabled) with an ERR_COR. If the Root Complex does not propagate the poison to host memory, it signals the error (if enabled) with ERR_NONFATAL.

Another example is a Requester that receives a poisoned Memory Read Completion TLP. If the Requester propagates the poisoned data internally or handles the error like it would for a Completion with UR/CA Status, it signals the error (if enabled) with an ERR_COR. If the Requester does not handle the poison in a manner that permits continued operation, it signals the error (if enabled) with ERR_NONFATAL.

6.2.3.2.4.4. Requester with Completion Timeout

This section applies to Requesters other than Root Ports performing programmed I/O (PIO). See Section 6.2.10.3 for related RP PIO functionality in Root Ports that support RP Extensions for DPC.

When the Requester of a Non-Posted Request times out while waiting for the associated Completion, the Requester is permitted to attempt to recover from the error by issuing a separate subsequent Request. The Requester is permitted to attempt recovery zero, one, or multiple (finite) times, but must signal the error (if enabled) with an uncorrectable error Message if no further recovery attempt will be made.

If the severity of the Completion Timeout is non-fatal, and the Requester elects to attempt recovery by issuing a new request, the Requester must first handle the current error case as an Advisory Non-Fatal Error.[77] A Requester with AER signals the error (if enabled) by sending an ERR_COR Message. A Requester without AER sends no error Message for this case.

[75] However, see Section 2.7.2.2 regarding certain Requests with Poisoned data that must be handled as uncorrectable errors.

[76] If the severity is fatal, the error is not an Advisory Non-Fatal Error, and must be signaled (if enabled) with ERR_FATAL.

[77] If the severity is fatal, the error is not an Advisory Non-Fatal Error, and must be signaled (if enabled) with ERR_FATAL. The Requester is strongly discouraged from attempting recovery since sending ERR_FATAL will often result in the entire hierarchy going down.

Note that automatic recovery by the Requester from a Completion Timeout is generally possible only if the Non-Posted Request has no side-effects, but may also depend upon other considerations outside the scope of this specification.

6.2.3.2.4.5. Receiver of an Unexpected Completion

When a Receiver receives an unexpected Completion and the severity of the Unexpected Completion error is non-fatal, the Receiver must handle this case as an Advisory Non-Fatal Error.[78] A Receiver with AER signals the error (if enabled) by sending an ERR_COR Message. A Receiver without AER sends no error Message for this case.

If the unexpected Completion was a result of misrouting, the Completion Timeout mechanism at the associated Requester will trigger eventually, and the Requester may elect to attempt recovery. Interference with Requester recovery can be avoided by having the Receiver of the unexpected Completion handle the error as an Advisory Non-Fatal Error.

6.2.3.2.5. Requester Receiving a Completion with UR/CA Status

When a Requester receives back a Completion with a UR/CA Status, generally the Completer has handled the error as an Advisory Non-Fatal Error, assuming the error severity was non-fatal at the Completer (see Section 6.2.3.2.4.1). The Requester must determine if any error recovery action is necessary, what type of recovery action to take, and whether or not to report the error.

If the Requester needs to report the error, the Requester must do so solely through a Requester-specific mechanism. For example, many devices have an associated device driver that can report errors to software. As another important example, the Root Complex on some platforms returns all 1's to software if a Configuration Read Completion has a UR/CA Status.

Section 6.2.10.3 covers RP PIO controls for Root Ports that support RP Extensions for DPC. Outside of the RP PIO mechanisms, ~~The~~ Requesters ~~is~~ are not permitted to report the error using PCI Express logging and error Message signaling.

Comment [VS190]: ECN: Enhanced DPC

6.2.3.3. Error Forwarding (Data Poisoning)

Error Forwarding, also known as data poisoning, is indicated by setting the EP bit in a TLP. Refer to Section 2.7.2. This is another method of error reporting in PCI Express that enables the Receiver of a TLP to associate an error with a specific Request or Completion. Unlike the Completion Status mechanism, Error Forwarding can be used with either Requests or Completions that contain data. In addition, "intermediate" Receivers along the TLP's route, not just the Receiver at the ultimate destination, are required to detect and report (if enabled) receiving the poisoned TLP. This can help software determine if a particular Switch along the path poisoned the TLP.

[78] If the severity is fatal, the error is not an Advisory Non-Fatal Error, and must be signaled (if enabled) with ERR_FATAL.

6.2.3.4. Optional Error Checking

This specification contains a number of optional error checks. Unless otherwise specified, behavior is undefined if an optional error check is not performed and the error occurs.

When an optional error check involves multiple rules, unless otherwise specified, each rule is independently optional. An implementation may check against all of the rules, none of them or any combination.

Unless otherwise specified, implementation specific criteria are used in determining whether an optional error check is performed.

6.2.4. Error Logging

Section 6.2.7 lists all the errors governed by this specification and for each error, the logging requirements are specified. Device Functions that do not support the Advanced Error Reporting Capability log only the Device Status register bits indicating that an error has been detected. Note that some errors are also reported using the reporting mechanisms in the PCI compatible (Type 00h and 01h) configuration registers. Section 7.5 describes how these register bits are affected by the different types of error conditions described in this section.

For device Functions supporting the Advanced Error Reporting Capability, each of the errors in Table 6-3, Table 6-4, and Table 6-5 corresponds to a particular bit in the Uncorrectable Error Status register or Correctable Error Status register. These registers are used by software to determine more precisely which error and what severity occurred. For specific Transaction Layer errors and Uncorrectable Internal Errors, the associated TLP header is recorded.

In a multi-Function device, PCI Express errors that are not related to any specific Function within the device, are logged in the corresponding status and logging registers of all Functions in that device.

The following PCI Express errors are not Function-specific:

- All Physical Layer errors
- All Data Link Layer errors
- These Transaction Layer errors:
 - ECRC Check Failed
 - Unsupported Request, when caused by no Function claiming a TLP
 - Receiver Overflow
 - Flow Control Protocol Error
 - Malformed TLP
 - Unexpected Completion, when caused by no Function claiming a Completion
 - Unexpected Completion, when caused by a Completion that cannot be forwarded by a Switch, and the Ingress Port is a Switch Upstream Port associated with a multi-Function device

Comment [VS191]: Errata B3

- Some Transaction Layer errors (e.g., Poisoned TLP Received) may be Function-specific or not, depending upon whether the associated TLP targets a single Function or all Functions in that device.

Comment [VS192]: Errata B3

- Some Internal Errors
 - The determination of whether an Internal Error is Function-specific or not is implementation specific.

On the detection of one of these errors, a multi-Function device should generate at most one error reporting Message of a given severity, where the Message must report the Requester ID of a Function of the device that is enabled to report that specific type of error. If no Function is enabled to send a reporting Message, the device does not send a reporting Message. If all reporting-enabled Functions have the same severity level set for the error, only one error Message is sent. If all reporting-enabled Functions do not have the same severity level set for the error, one error Message for each severity level is sent. Software is responsible for scanning all Functions in a multi-Function device when it detects one of those errors.

6.2.4.1. Root Complex Considerations (Advanced Error Reporting)

6.2.4.1.1. Error Source Identification

In addition to the above logging, a Root Port or Root Complex Event Collector that supports the Advanced Error Reporting Capability is required to implement the Error Source Identification register, which records the Requester ID of the first ERR_NONFATAL/ERR_FATAL (uncorrectable errors) and ERR_COR (correctable errors) Messages received by the Root Port or Root Complex Event Collector. System software written to support Advanced Error Reporting can use the Root Error Status register to determine which fields hold valid information.

If a Root Complex Event Collector is implemented, errors from a Root Complex Integrated Endpoint may optionally be reported in a Root Complex Event Collector residing on the same Logical Bus as the Root Complex Integrated Endpoint. The Root Complex Event Collector must explicitly declare supported Root Complex Integrated Endpoints as part of its capabilities. Each Root Complex Integrated Endpoint must be associated with no more than one Root Complex Event Collector.

For both Root Ports and Root Complex Event Collectors, in order for a received error Message or an internally generated error Message to be recorded in the Root Error Status register and the Error Source Identification register, the error Message must be "transmitted." Refer to Section 6.2.8.1 for information on how received Messages are forwarded and transmitted. Internally generated error Messages are enabled for transmission with the SERR# Enable bit in the Command register (ERR_NONFATAL and ERR_FATAL) or the Reporting Enable bits in the Device Control register (ERR_COR, ERR_NONFATAL, and ERR_FATAL).

6.2.4.1.2. Interrupt Generation

The Root Error Command register allows further control of Root Complex response to Correctable, Non-Fatal, and Fatal error Messages than the basic Root Complex capability to generate system errors in response to error Messages. Bit fields enable or disable generation of interrupts for the

three types of error Messages. System error generation in response to error Messages may be disabled via the PCI Express Capability structure.

If a Root Port or Root Complex Event Collector is enabled for level-triggered interrupt signaling using the INTx messages, the virtual INTx wire must be asserted whenever and as long as all of the following conditions are satisfied:

- The Interrupt Disable bit in the Command register is set to 0b.
- At least one Error Reporting Enable bit in the Root Error Command register and its associated error Messages Received bit in the Root Error Status register are both set to 1b.

Note that all other interrupt sources within the same Function will assert the same virtual INTx wire when requesting service.

If a Root Port or Root Complex Event Collector is enabled for edge-triggered interrupt signaling using MSI or MSI-X, an interrupt message must be sent every time the logical AND of the following conditions transitions from FALSE to TRUE:

- The associated vector is unmasked (not applicable if MSI does not support PVM).
- At least one Error Reporting Enable bit in the Root Error Command register and its associated error Messages Received bit in the Root Error Status register are both set to 1b.

Note that Advanced Error Reporting MSI/MSI-X interrupts always use the vector indicated by the Advanced Error Interrupt Message Number field in the Root Error Status register.

6.2.4.2. Multiple Error Handling (Advanced Error Reporting Capability)

For the Advanced Error Reporting Capability, the Uncorrectable Error Status register and Correctable Error Status register accumulate the collection of errors which correspond to that particular PCI Express interface. The bits remain set until explicitly cleared by software or reset. Since multiple bits might be set in the Uncorrectable Error Status register, the First Error Pointer ~~register~~ (when valid) points to the oldest ~~unmasked~~ uncorrectable error that is recorded~~occurred first~~. The First Error Pointer ~~register~~ is valid when the corresponding bit of the Uncorrectable Error Status register is set. The First Error Pointer ~~value~~ is invalid~~not meaningful~~ when the corresponding bit of the Uncorrectable Error Status register is not set, or is an ~~unimplemented or~~ undefined bit.

Comment [VS193]: B15

The Advanced Error Reporting Capability provides the ability to record headers[79] for errors that require header logging. An implementation may support the recording of multiple headers, but at a minimum must support the ability of recording at least one. The ability to record multiple headers is indicated by the state of the Multiple Header Recording Capable bit and enabled by the Multiple Header Recording Enable bit of the Advanced Error Capabilities and Control register. When multiple header recording is supported and enabled, ~~headers~~ errors are recorded in the order in which ~~the corresponding errors~~they are detected.

79 If a Function supports TLP Prefixes, then its AER Capability also records any accompanying TLP Prefix along with each recorded header. References to header recording also imply TLP Prefix recording.

~~When the First Error Pointer register is valid, the Header Log register contains the recorded header for the corresponding error. Writing a 1b to the bit of the Uncorrectable Error Status register to which a valid First Error Pointer register corresponds causes the space occupied by that recorded header to be released, the next recorded header to be reported in the Header Log register, and the First Error Pointer register to be updated to point to the corresponding Uncorrectable Error Status register bit. If no further recorded header is of the same error type as the last, then the bit in the Uncorrectable Error Status register to which the 1b was written is cleared; otherwise, it remains Set. When the last recorded header is reached and a 1b is written to the corresponding Uncorrectable Error Status register bit, then the First Error Pointer register becomes invalid (i.e., corresponds to an Uncorrectable Error Status register bit that is not Set, undefined, or unimplemented) and the value of the Header Log register is undefined.~~

Comment [VS194]: Errata B15

If no header recording resources are available when an unmasked uncorrectable error is detected, its error status bit is set, but the error is not recorded. If an uncorrectable error is masked when it is detected, its error status bit is set, but the error is not recorded.

When software is ready to dismiss a recorded error indicated by the First Error Pointer, software writes a 1b to the indicated error status bit to clear it, which causes hardware to free up the associated recording resources. If another instance of that error is still recorded, hardware is permitted but not required to leave that error status bit set. If any error instance is still recorded, hardware must immediately update the Header Log, TLP Prefix Log, TLP Prefix Log Present bit, First Error Pointer, and Uncorrectable Error Status register to reflect the next recorded error. If no other error is recorded, it is recommended that hardware update the First Error Pointer to indicate a status bit that it will never set, e.g., a Reserved status bit. See the Implementation Note below.

If multiple header recording is supported and enabled, and the First Error Pointer is valid, it is recommended that software not write a 1b to any status bit other than the one indicated by the First Error Pointer[80]. If software writes a 1b to such non-indicated bits, hardware is permitted to clear any associated recorded errors, but is not required to do so.

If software observes that the First Error Pointer is invalid, and software wishes to clear any unmasked status bits that were set because of earlier header recording resource overflow, software should be aware of the following race condition. If any new instances of those errors happen to be recorded before software clears those status bits, one or more of the newly recorded errors might be lost.

If multiple header recording is supported and enabled, software must use special care when clearing the Multiple Header Recording Enable bit. Hardware behavior is undefined if software clears that bit while the First Error Pointer is valid. Before clearing the Multiple Header Recording Enable bit, it is recommended that software temporarily mask all uncorrectable errors, and then repetitively dismiss each error indicated by the First Error Pointer.

Since an implementation only has the ability to record a finite number of headers, it is important that software services the First Error Pointer, Header Log, and TLP Prefix Log registers in a timely manner, to limit the risk of missing this information for subsequent errors. A Header Log Overflow occurs when an error that requires header logging is detected and either the number of recorded

[80] Status bits for masked errors are an exception. Software can safely clear them if software is certain that they have no recorded headers, as would be the case if they have remained masked since the First Error Pointer was last invalid.

headers supported by an implementation has been reached, or the Multiple Header Recording Enable bit is not Set and the First Error Pointer ~~register~~ is valid.

Implementations may optionally check for this condition and report a Header Log Overflow error. This is a reported error associated with the detecting Function.

The setting of Multiple ~~Heading~~ Header Recording Capable and the checking for Header Log Overflow are independently optional.

Comment [VS195]: Errata B15

Comment [VS196]: B15

Formatted: Font color: Black

IMPLEMENTATION NOTE

First Error Pointer Register Being Valid

The First Error Pointer (FEP) field is defined to be valid when the corresponding bit of the Uncorrectable Error Status register is set. To avoid ambiguity with certain cases, the following is recommended:

- After an uncorrectable error has been recorded, when the associated bit in the Uncorrectable Error Status register is cleared by software writing a 1b to it, hardware should update the FEP to point to a status bit that it will never set, e.g., a Reserved status bit. (This assumes that the Function does not already have another recorded error to report, as could be the case if it supports multiple header recording.)
- The default value for the FEP should point to a status bit that hardware will never set, e.g., a Reserved status bit.

Here is an example case of ambiguity with Unsupported Request (UR) if the above recommendations are not followed:

- UR and Advisory Non-Fatal Error are unmasked while system firmware does its Configuration Space probing.
- The Function encounters a UR due to normal probing, logs it, and sets the FEP to point to UR.
- System firmware clears the UR Status bit, and hardware leaves the FEP pointing to UR.
- After the OS has booted, it masks UR.
- Normal probing sets the UR Status bit, but the error is not recorded since UR is masked.

At this point, there's the ambiguity of the FEP pointing to a status bit that is set (thus being valid), when in fact, there is no recorded error that needs to be processed by software.

If hardware relies on this definition of the FEP being valid to determine when it's possible to record a new error, the Function can fail to record new unmasked errors, falsely determining that it has no available recording resources. Hardware implementations that rely on other internal state to determine when it's possible to record a new error might not have this problem; however, hardware implementations should still follow the above recommendations to avoid presenting this ambiguity to software.

6.2.4.3. Advisory Non-Fatal Error Logging

Section 6.2.3.2.4 describes Advisory Non-Fatal Error cases, under which an agent with AER detecting an uncorrectable error of non-fatal severity signals the error (if enabled) using ERR_COR instead of ERR_NONFATAL. For the same cases, an agent without AER sends no error Message. The remaining discussion in this section is in the context of agents that do implement AER.

For Advisory Non-Fatal Error cases, since an uncorrectable error is signaled using the correctable error Message, control/status/mask bits involving both uncorrectable and correctable errors apply. Figure 6-2 shows a flowchart of the sequence. Following are some of the unique aspects for logging Advisory Non-Fatal Errors.

First, the uncorrectable error needs to be of severity non-fatal, as determined by the associated bit in the Uncorrectable Error Severity register. If the severity is fatal, the error does not qualify as an Advisory Non-Fatal Error, and will be signaled (if enabled) with ERR_FATAL.

Next, the specific error case needs to be one of the Advisory Non-Fatal Error cases documented in Section 6.2.3.2.4. If not, the error does not qualify as an Advisory Non-Fatal Error, and will be signaled (if enabled) with an uncorrectable error Message.

Next, the Advisory Non-Fatal Error Status bit is set in the Correctable Error Status register to indicate the occurrence of the advisory error, and the Advisory Non-Fatal Error Mask bit in the Correctable Error Mask register is checked, and, if set, no further processing is done.

If the Advisory Non-Fatal Error Mask bit is clear, logging proceeds by setting the "corresponding" bit in the Uncorrectable Error Status register, based upon the specific uncorrectable error that's being reported as an advisory error. If the "corresponding" uncorrectable error bit in the Uncorrectable Error Mask register is clear and the error is one that requires header logging, then the prefix and header are recorded, subject to the availability of resources. See Section 6.2.4.2.

Finally, an ERR_COR Message is sent if the Correctable Error Reporting Enable bit is set in the Device Control register.

6.2.4.4. TLP Prefix Logging

For any device Function that supports both TLP Prefixes and Advanced Error Reporting the TLP Prefixes associated with the TLP in error are recorded in the TLP Prefix Log register according to the same rules as the Header Log register (such that both the TLP Prefix Log and Header Log registers always correspond to the error indicated in the First Error Pointer ~~register~~, when the First Error Pointer ~~register~~ is valid).

Comment [VS197]: B15

The TLP Prefix Log Present bit (see Section 7.10.7) indicates that the TLP Prefix Log register (see Section 7.10.12) contains information.

Only End-End TLP Prefixes are logged by AER. Logging of Local TLP Prefixes may occur elsewhere using prefix specific mechanisms.[81]

End-End TLP Prefixes are logged in the TLP Prefix Log register. The underlying TLP Header is logged in the Header Log register subject to two exceptions:

[81] For example, errors involving MRI-IOV TLP Prefixes are logged in MR-IOV structures and are not logged in the AER Capability.

- If the Extended Fmt Field Supported bit is Set (see Section 7.8.15), a Function that does not support TLP Prefixes and receives a TLP containing a TLP Prefix will signal Malformed TLP and the Header Log register will contain the first four DWs of the TLP (TLP Prefixes followed by as much of the TLP Header as will fit).
- A Function that receives a TLP containing more End-End TLP Prefixes than are indicated by the Function's Max End-End TLP Prefixes field must handle the TLP as an error (see Section 2.2.10.2 for specifics) and store the first overflow End-End TLP Prefix in the 1st DW of the Header Log register with the remainder of the Header Log register being undefined.

6.2.5. Sequence of Device Error Signaling and Logging Operations

Figure 6-2 shows the sequence of operations related to signaling and logging of errors detected by a device.

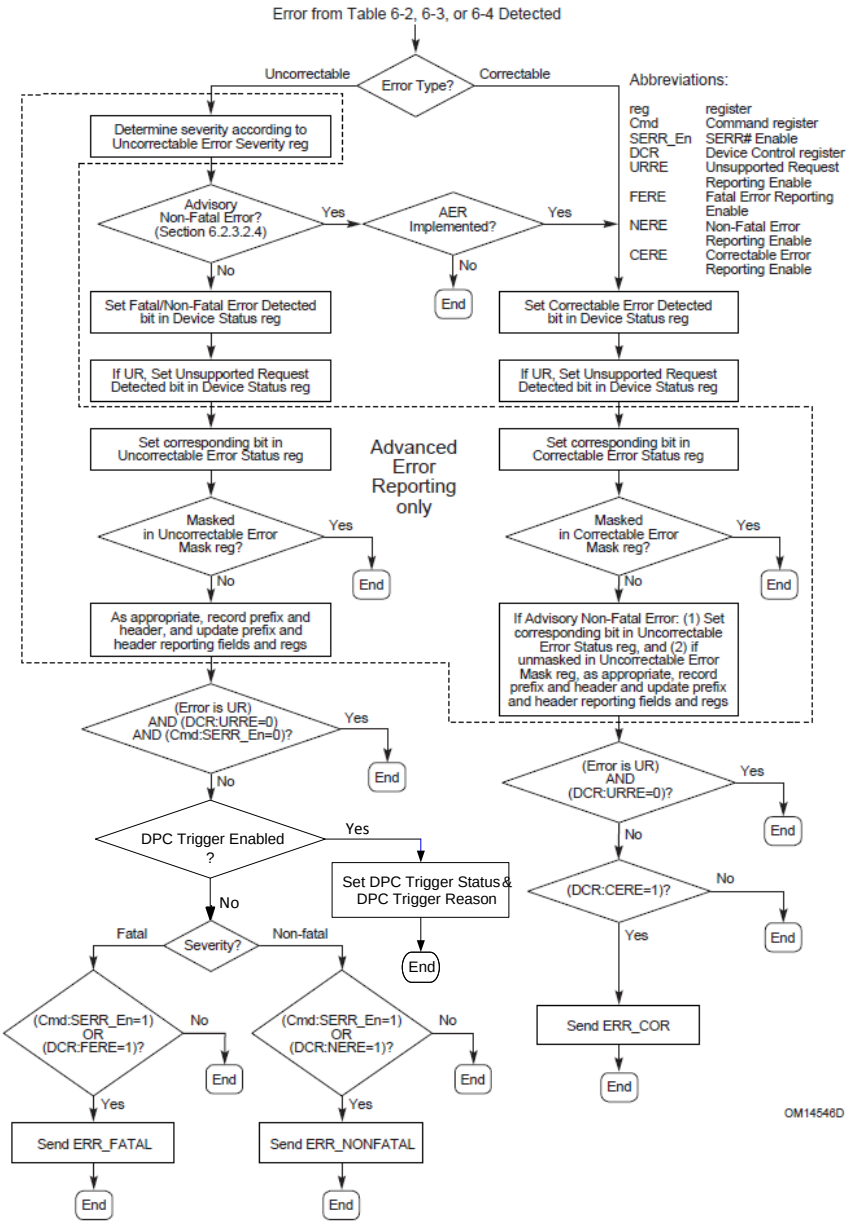


Figure 6-2: Flowchart Showing Sequence of Device Error Signaling and Logging Operations

6.2.6. Error Message Controls

Error Messages have a complex set of associated control and status bits. Figure 6-3 provides a conceptual summary in the form of a pseudo logic diagram for how error Messages are generated, logged, forwarded, and ultimately notified to the system. Not all logged status bits are shown. The logic gates shown in this diagram are intended for conveying general concepts, and not for direct implementation.

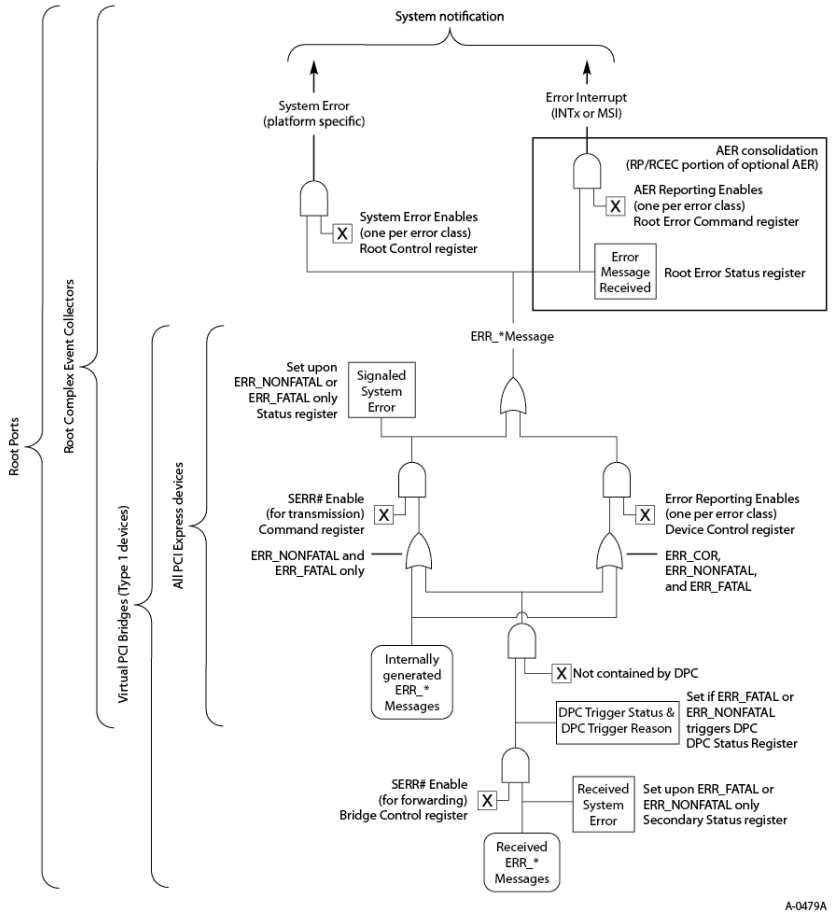


Figure 6-3: Pseudo Logic Diagram for Error Message Controls

Comment [VS198]: Errata B20

6.2.7. Error Listing and Rules

Table 6-2 through Table 6-4 list all of the PCI Express errors that are defined by this specification. Each error is listed with a short-hand name, how the error is detected in hardware, the default severity of the error, and the expected action taken by the agent which detects the error. These actions form the rules for PCI Express error reporting and logging.

The Default Severity column specifies the default severity for the error without any software reprogramming. For device Functions supporting the Advanced Error Reporting Capability, the uncorrectable errors are programmable to Fatal or Non-fatal with the Error Severity register. Device Functions without Advanced Error Reporting Capability use the default associations and are not reprogrammable.

The detecting agent action for Downstream Ports that implement Downstream Port Containment (DPC) and have it enabled will be different if the error triggers DPC. DPC behavior is not described in the following tables. See Section 6.2.10 for the description of DPC behavior.

Comment [VS199]: Errata B21.

Table 6-2: General PCI Express Error List

Error Name	Error Type (Default Severity)	Detecting Agent Action[82]	References
Corrected Internal Error	Correctable (masked by default)	Component: Send ERR_COR to Root Complex.	Section 6.2.9
Uncorrectable Internal Error	Uncorrectable (Fatal and masked by default)	*Component:* Send ERR_FATAL to Root Complex. Optionally, log the prefix/header of the first TLP associated with the error.	Section 6.2.9
Header Log Overflow	Correctable (masked by default)	Component: Send ERR_COR to Root Complex.	Section 6.2.4.2

[82] For these tables, detecting agent action is given as if all enable bits are set to "enable" and, for Advanced Error Handling, mask bits are disabled and severity bits are set to their default values. Actions must be modified according to the actual settings of these bits.

Table 6-3: Physical Layer Error List

Error Name	Error Type (Default Severity)	Detecting Agent Action[83]	References
Receiver Error	Correctable	Receiver: Send ERR_COR to Root Complex.	Section 4.2.1.1.3 Section 4.2.1.2 Section 4.2.4.7 Section 4.2.6

Table 6-4: Data Link Layer Error List

Error Name	Error Type (Default Severity)	Detecting Agent Action[81]	References
Bad TLP	Correctable	Receiver: Send ERR_COR to Root Complex.	Section 3.5.3.1
Bad DLLP		Receiver: Send ERR_COR to Root Complex.	Section 3.5.2.2
Replay Timer Timeout		Transmitter: Send ERR_COR to Root Complex.	Section 3.5.2.1
REPLAY_NUM Rollover		Transmitter: Send ERR_COR to Root Complex.	Section 3.5.2.1
Data Link Protocol Error	Uncorrectable (Fatal)	If checking, send ERR_FATAL to Root Complex.	Section 3.5.2.2
Surprise Down		If checking, send ERR_FATAL to Root Complex.	Section 3.2.1

[83] For these tables, detecting agent action is given as if all enable bits are set to "enable" and, for Advanced Error Handling, mask bits are disabled and severity bits are set to their default values. Actions must be modified according to the actual settings of these bits.

Table 6-5: Transaction Layer Error List

Error Name	Error Type (Default Severity)	Detecting Agent Action[81]	References
Poisoned TLP Received	Uncorrectable (Non-Fatal)	Receiver: Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error cases described in Sections 6.2.3.2.4.2 and 6.2.3.2.4.3. Log the prefix/header of the Poisoned TLP.[84]	Section 2.7.2.2
Poisoned TLP Egress Blocked		Downstream Port Transmitter: Send ERR_NONFATAL to Root Complex. Log the prefix/header of the poisoned TLP.	Section 2.7.2.2
ECRC Check Failed		Receiver (if ECRC checking is supported): Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error case described in Section 6.2.3.2.4.2. Log the prefix/header of the TLP that encountered the ECRC error.	Section 2.7.1
Unsupported Request (UR)		Request Receiver: Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error case described in Section 6.2.3.2.4.1. Log the prefix/header of the TLP that caused the error.	Section 2.2.8.6, Section 2.3.1, Section 2.3.2, Section 2.7.2.2, Section 2.9.1, Section 5.3.1, Section 6.2.3.1, Section 6.2.6, Section 6.2.8.1, Section 6.5.7, Section 7.3.1, Section 7.3.3, Section 7.5.1.1, Section 7.5.1.2

Comment [VS200]: ECN: Enhanced Downstream Port containment (eDPC)

[84] Advanced Error Handling only.

Error Name	Error Type (Default Severity)	Detecting Agent Action[81]	References
Completion Timeout	Uncorrectable (Non-Fatal)	Requester: Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error case described in Section 6.2.3.2.4.4. If the Completion Timeout Prefix/Header Log Capable bit is Set in the Advanced Error Capabilities and Control register, log the prefix/header of the Request TLP that encountered the error.	Section 2.8
Completer Abort		Completer: Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error case described in Section 6.2.3.2.4.1. Log the prefix/header of the Request that encountered the error.	Section 2.3.1
Unexpected Completion		Receiver: Send ERR_COR to Root Complex. This is an Advisory Non-Fatal Error case described in Section 6.2.3.2.4.5. Log the prefix/header of the Completion that encountered the error.	Section 2.3.2
ACS Violation		Receiver (if checking): Send ERR_NONFATAL to Root Complex. Log the prefix/header of the Request TLP that encountered the error.	
MC Blocked TLP		*Receiver (if checking):* Send ERR_NONFATAL to Root Complex. Log the prefix/header of the Request TLP that encountered the error.	Section 6.14.4

Comment [VS201]: ECN: Downstream Port containment (DPC)

Error Name	Error Type (Default Severity)	Detecting Agent Action[81]	References
AtomicOp Egress Blocked	Uncorrectable (Non-Fatal)	Egress Port: Send ERR_COR to Root Complex. This is an Advisory Non-Fatal Error case described in Section 6.2.3.2.4.1. Log the prefix/header of the AtomicOp Request that encountered the error.	Section 6.15.2
TLP Prefix Blocked		Egress Port: Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error case described in Section 6.2.3.2.4.1. Log the prefix/header of the TLP that encountered the error.	Section 2.2.10.2
Receiver Overflow	Uncorrectable (Fatal)	Receiver (if checking): Send ERR_FATAL to Root Complex.	Section 2.6.1.2
Flow Control Protocol Error		Receiver (if checking): Send ERR_FATAL to Root Complex.	Section 2.6.1
Malformed TLP		Receiver: Send ERR_FATAL to Root Complex. Log the prefix/header of the TLP that encountered the error.	Section 2.2.2, Section 2.2.3, Section 2.2.5, Section 2.2.7, Section 2.2.8.1, Section 2.2.8.2, Section 2.2.8.3, Section 2.2.8.4, Section 2.2.8.5, Section 2.2.8.10, Section 2.2.10, Section 2.2.10.1, Section 2.2.10.2, Section 2.3, Section 2.3.1, Section 2.3.1.1, Section 2.3.2, Section 2.5, Section 2.5.3, Section 2.6.1, Section 2.6.1.2, Section 6.2.4.4, Section 6.3.2
~~TLP Prefix Blocked~~		~~Egress Port:~~ ~~Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error case described in Section 6.2.3.2.4.1.~~ ~~Log the prefix/header of the TLP that encountered the error.~~	~~Section 2.2.10.2~~

Comment [VS202]: Errata B7

Formatted Table

For all errors listed above, the appropriate status bit(s) must be set upon detection of the error. For Unsupported Request (UR), additional detection and reporting enable bits apply (see Section 6.2.5).



IMPLEMENTATION NOTE

Device UR Reporting Compatibility with Legacy and 1.0a Software

With 1.0a device Functions that do not implement Role-Based Error Reporting,[85] the Unsupported Request Reporting Enable bit in the Device Control register, when clear, prevents the Function from sending any error Message to signal a UR error. With Role-Based Error Reporting Functions, if the SERR# Enable bit in the Command register is set, the Function is implicitly enabled[86] to send ERR_NONFATAL or ERR_FATAL messages to signal UR errors, even if the Unsupported Request Reporting Enable bit is clear. This raises a backward compatibility concern with software (or firmware) written for 1.0a devices.

With software/firmware that sets the SERR# Enable bit but leaves the Unsupported Request Reporting Enable and Correctable Error Reporting Enable bits clear, a Role-Based Error Reporting Function that encounters a UR error will send no error Message if the Request was non-posted, and will signal the error with ERR_NONFATAL if the Request was posted. The behavior with non-posted Requests supports PC-compatible Configuration Space probing, while the behavior with posted Requests restores error reporting compatibility with PCI and PCI-X, avoiding the potential in this area for silent data corruption. Thus, Role-Based Error Reporting devices are backward compatible with envisioned legacy and 1.0a software and firmware.

6.2.7.1. Conventional PCI Mapping

In order to support conventional PCI driver and software compatibility, PCI Express error conditions, where appropriate, must be mapped onto the PCI Status register bits for error reporting.

In other words, when certain PCI Express errors are detected, the appropriate PCI Status register bit is set alerting the error to legacy PCI software. While the PCI Express error results in setting the PCI Status register, clearing the PCI Status register will not result in clearing bits in the Uncorrectable Error Status register and Correctable Error Status register. Similarly, clearing bits in the Uncorrectable Error Status register and Correctable Error Status register will not result in clearing the PCI Status register.

The PCI command register has bits which control PCI error reporting. However, the PCI Command register does not affect the setting of the PCI Express error register bits.

6.2.8. Virtual PCI Bridge Error Handling

Virtual PCI Bridge configuration headers are associated with each PCI Express Port in a Root Complex or a Switch. For these cases, PCI Express error concepts require appropriate mapping to the PCI error reporting structures.

[85] As indicated by the Role-Based Error Reporting bit in the Device Capabilities register. See Section 7.8.3.

[86] Assuming the Unsupported Request Error Mask bit is not set in the Uncorrectable Error Mask register if the device implements AER.

6.2.8.1. Error Message Forwarding and PCI Mapping for Bridge - Rules

In general, a TLP is either passed from one side of the Virtual PCI Bridge to the other, or is handled at the ingress side of the Bridge according to the same rules which apply to the ultimate recipient of a TLP. The following rules cover PCI Express specific error related cases. Refer to Section 6.2.6 for a conceptual summary of Error Message Controls.

- If a Request does not address a space mapped to either the Bridge's internal space, or to the egress side of the Bridge, the Request is terminated at the ingress side as an Unsupported Request
- Poisoned TLPs are forwarded according to the same rules as non-Poisoned TLPs
 - When forwarding a Poisoned Request Downstream:
 - Set the Detected Parity Error bit in the Status register
 - Set the Master Data Parity Error bit in the Secondary Status register if the Parity Error Response Enable bit in the Bridge Control register is set
 - When forwarding a Poisoned Completion Downstream:
 - Set the Detected Parity Error bit in the Status register
 - Set the Master Data Parity Error bit in the Status register if the Parity Error Response bit in the Command register is set
 - When forwarding a Poisoned Request Upstream:
 - Set the Detected Parity Error bit in the Secondary Status register
 - Set the Master Data Parity Error bit in the Status register if the Parity Error Response bit in the Command register is set
 - When forwarding a Poisoned Completion Upstream:
 - Set the Detected Parity Error bit in the Secondary Status register
 - Set the Master Data Parity Error bit in the Secondary Status register if the Parity Error Response Enable bit in the Bridge Control register is set
- ERR_COR, ERR_NONFATAL, and ERR_FATAL are forwarded from the secondary interface to the primary interface, if the SERR# Enable bit in the Bridge Control register is set. A Bridge forwarding an error Message must not set the corresponding Error Detected bit in the Device Status register. Transmission of forwarded error Messages by the primary interface is controlled by multiple bits, as shown in Figure 6-3.
- For a Root Port, error Messages forwarded from the secondary interface to the primary interface must be enabled for "transmission" by the primary interface in order to cause a System Error via the Root Control register or (when the Advanced Error Reporting Capability is present) reporting via the Root Error Command register and logging in the Root Error Status register and Error Source Identification register.
- For a Root Complex Event Collector (technically not a Bridge), error Messages "received" from associated Root Complex Integrated Endpoints must be enabled for "transmission" in order to

cause a System Error via the Root Control register or (when the Advanced Error Reporting Capability is present) reporting via the Root Error Command register and logging in the Root Error Status register and Error Source Identification register.

6.2.9. Internal Errors

An Internal Error is an error associated with a PCI Express interface that occurs within a component and which may not be attributable to a packet or event on the PCI Express interface itself or on behalf of transactions initiated on PCI Express. The determination of what is considered an Internal Error is implementation specific and is outside the scope of this specification.

Internal Errors may be classified as Corrected Internal Errors or Uncorrectable Internal Errors. A Corrected Internal Error is an error that occurs within a component that has been masked or worked around by hardware without any loss of information or improper operation. An example of a possible Corrected Internal Error is an internal packet buffer memory error corrected by an Error Correcting Code (ECC). An Uncorrectable Internal Error is an error that occurs within a component that results in improper operation of the component. An example of a possible Uncorrectable Internal Error is a memory error that cannot be corrected by an ECC. The only method of recovering from an Uncorrectable Internal Error is reset or hardware replacement.

Reporting of Corrected Internal Errors and Uncorrectable Internal Errors is independently optional. If either is reported, then AER must be implemented.

Header logging is optional for Uncorrectable Internal Errors. When a header is logged, the header is that of the first TLP that was lost or corrupted by the Uncorrectable Internal Error. When header logging is not implemented or a header is not available, a header of all ones is recorded.

Internal Errors that can be associated with a specific PCI Express interface are reported by the Function(s) associated with that Port. Internal Errors detected within Switches that cannot be associated with a specific PCI Express interface are reported by the Upstream Port. Reporting of Internal Errors that cannot be associated with a specific PCI Express interface in all other multi-Port components (e.g., Root Complexes) is outside the scope of this specification.

6.2.10. Downstream Port Containment (DPC)

Comment [VS203]: ECN: Downstream Port containment (DPC)

Downstream Port Containment (DPC) is an optional normative feature of a Downstream Port. DPC halts PCI Express traffic below a Downstream Port after an unmasked uncorrectable error is detected at or below the Port, avoiding the potential spread of any data corruption, and permitting error recovery if supported by software. A Downstream Port indicates support for DPC by implementing a DPC Extended Capability structure, which contains all DPC control and status bits. See Section 7.31.

DPC is disabled by default, and cannot be triggered unless enabled by software using the DPC Trigger Enable field. When the DPC Trigger Enable field is set to 01b, DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_FATAL Message. When the DPC Trigger Enable field is set to 10b, DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_NONFATAL or ERR_FATAL Message. In addition to uncorrectable errors of the type managed by the PCI Express Extended Capability and Advanced Error Reporting (AER), RP PIO errors can be handled as

uncorrectable errors. See Section 6.2.10.3. There is also a mechanism described in Section 6.2.10.4 for software or firmware to trigger DPC.

When DPC is triggered due to receipt of an uncorrectable error Message, the Requester ID from the Message is recorded in the DPC Error Source ID register and that Message is discarded and not forwarded Upstream. When DPC is triggered by an unmasked uncorrectable error, that error will not be signaled with an uncorrectable error Message, even if otherwise enabled. However, when DPC is triggered, DPC can signal an interrupt or send an ERR_COR Message if enabled. See Sections 6.2.10.1 and 6.2.10.2.

When DPC is triggered, the Downstream Port immediately Sets the DPC Trigger Status bit and DPC Trigger Reason field to indicate the triggering condition (unmasked uncorrectable error, ERR_NONFATAL, ERR_FATAL, RP_PIO error, or software triggered), and disables its Link by directing the LTSSM to the Disabled state. Once the LTSSM reaches the Disabled state, it remains in that state until the DPC Trigger Status bit is Cleared. To ensure that the LTSSM has time to reach the Disabled state or at least to bring the Link down under a variety of error conditions, software must leave the Downstream Port in DPC until the Data Link Layer Link Active bit in the Link Status register reads 0b. See Section 7.8.8. See Section 2.9.3 for other important details on Transaction Layer behavior during DPC.

After DPC has been triggered in a Root Port that supports RP Extensions for DPC, the Root Port may require some time to quiesce and clean up its internal activities, such as those associated with DMA read Requests. When the DPC Trigger Status bit is Set and the DPC RP Busy bit is Set, software must leave the Root Port in DPC until the DPC RP Busy bit reads 0b.

After software releases the Downstream Port from DPC, the associated Link will normally attempt to retrain. Software can use Data Link Layer State Changed interrupts, DL_Active ERR_COR signaling, or both, to signal when the Link reaches the DL_Active state again. See Sections 6.7.3.3 and 6.2.10.5.



IMPLEMENTATION NOTE

Data Value of All 1's

Many platforms, including those supporting RP Extensions for DPC, can return a data value of all 1's to software when an error is associated with a PCI Express Configuration, I/O, or Memory Read Request. During DPC, the Downstream Port discards Requests destined for the Link and completes them with an error (i.e., either with an Unsupported Request (UR) or Completer Abort (CA) Completion Status). By ending a series of MMIO or configuration space operations with a read to an address with a known data value not equal to all 1's, software may determine if a Completer has been removed or DPC has been triggered.

Also see the Implementation Note "Use of RP PIO Advisory Non-Fatal Errors"

Formatted: Font color: Black

Formatted: Font color: Black

IMPLEMENTATION NOTE

Selecting Non-Posted Request Response During DPC

The DPC Completion Control bit determines how a Downstream Port responds to a Non-Posted Request (NPR) received during DPC. The selection needs to take into account how the rest of the platform handles PCI Express uncorrectable error recovery.

While specific PCI Express uncorrectable error recovery mechanisms in a platform are outside the scope of this specification, here are some guidelines based on general considerations.

If the platform or drivers do not support a PCI Express uncorrectable error recovery strategy, there's no envisioned benefit to enabling DPC, and thus no need to select the NPR response.

If the PCI Express uncorrectable error recovery strategy relies on software detecting containment by looking for all 1's returned by PIO reads, then a UR Completion may be the more appropriate selection, assuming the RP synthesizes an all 1's return value for PIO reads that return UR Completions. The all 1's synthesis would need to occur for PIO reads that target Configuration Space, Memory Space, and perhaps I/O Space.

If the PCI Express uncorrectable error recovery strategy utilizes a mechanism that handles UR and CA Completions differently for PIO reads, then a CA Completion might be the more appropriate selection. CA Completions coming back from a PCI Express device normally indicate a device programming model violation, which may need to trigger Root containment and error recovery.

Formatted: Font color: Black

IMPLEMENTATION NOTE

Selecting the DPC Trigger Condition

Non-Fatal Errors are uncorrectable errors that indicate that a particular TLP was unreliable, and in general the associated Function should not continue its normal operation. Fatal errors are uncorrectable errors that indicate that a particular Link and its related hardware are unreliable, and in general the entire hierarchy below that Link should not continue normal operation. This distinction between Non-Fatal and Fatal errors together with the Root Port error containment capabilities can sometimes be used to select the appropriate DPC trigger condition. The following assumes that there is no peer-to-peer traffic between devices.

Some RCs implement a proprietary feature that will be referred to generically as "Function Level Containment" (FLC). This is not an architected feature of PCI Express. A Root Port that implements FLC is capable of containing the traffic associated with a specific Function when a Non-Fatal Error is detected in that traffic. Switch Downstream Ports below a Root Port with FLC should be configured to trigger DPC when the Downstream Port detects an unmasked uncorrectable error itself or when the Downstream Port receives an ERR_FATAL Message. Under this mode, the Switch Downstream Port passes ERR_NONFATAL Messages it receives Upstream without triggering DPC. This enables Root Port FLC to handle Non-Fatal Errors that render a specific Function unreliable and Switch Downstream Port DPC to handle errors that render a sub-tree of the hierarchy domain unreliable. The Downstream Port still needs to trigger DPC for all unmasked uncorrectable errors it detects, since an ERR_NONFATAL it generates will have its own Requester ID, and the FLC hardware in the Root Port would not be able to determine which specific Function below the Switch Downstream Port was responsible for the Non-Fatal Error.

Switch Downstream Ports below a Root Port without FLC should be configured to trigger DPC when the Switch Downstream Port detects an unmasked uncorrectable error or when the Switch Downstream Port receives an ERR_NONFATAL or ERR_FATAL Message. This enables DPC to contain the error to the affected hierarchy below the Link and allow continued normal operation of the unaffected portion of the hierarchy domain.

Formatted: Font color: Purple

IMPLEMENTATION NOTE

Software Polling the DPC RP Busy Bit

The DPC RP Busy bit is a means for hardware to indicate to software that the RP needs to remain in DPC containment while the RP does some internal cleanup and quiescing activities. While the details of these activities are implementation specific, the activities will typically complete within a few microseconds or less. However, under worst-case conditions such as those that might occur with certain internal errors in large systems, the busy period might extend substantially, possibly into multiple seconds. If software is unable to tolerate such lengthy delays within the current software context, software may need to rely on using timer interrupts to schedule polling under interrupt.

Formatted: Font color: Auto

IMPLEMENTATION NOTE

Determination of DPC Control

DPC may be controlled in some configurations by platform firmware and in other configurations by the operating system. DPC functionality is strongly linked with the functionality in Advanced Error Reporting. To avoid conflicts over whether platform firmware or the operating system have control of DPC, it is recommended that platform firmware and operating systems always link the control of DPC to the control of Advanced Error Reporting.

Comment [VS204]: ECN: Downstream Port containment (DPC)

6.2.10.1. DPC Interrupts

A DPC-capable Downstream Port must support the generation of DPC interrupts. DPC interrupts are enabled by the DPC Interrupt Enable bit in the DPC Control register. DPC interrupts are indicated by the DPC Interrupt Status bit in the DPC Status register.

If the Port is enabled for level-triggered interrupt signaling using INTx messages, the virtual INTx wire must be asserted whenever and as long as the following conditions are satisfied:

- The value of the Interrupt Disable bit in the Command register is 0b.
- The value of the DPC Interrupt Enable bit is 1b.
- The value of the DPC Interrupt Status bit is 1b.

Note that all other interrupt sources within the same Function will assert the same virtual INTx wire when requesting service.

If the Port is enabled for edge-triggered interrupt signaling using MSI or MSI-X, an interrupt message must be sent every time the logical AND of the following conditions transitions from FALSE to TRUE:

- The associated vector is unmasked (not applicable if MSI does not support PVM).
- The value of the DPC Interrupt Enable bit is 1b.
- The value of the DPC Interrupt Status bit is 1b.

The Port may optionally send an interrupt message if interrupt generation has been disabled, and the logical AND of the above conditions is TRUE when interrupt generation is subsequently enabled.

The interrupt message will use the vector indicated by the DPC Interrupt Message Number field in the DPC Capability register. This vector may be the same or may be different from the vectors used by other interrupt sources within this Function.

6.2.10.2. DPC ERR_COR Signaling

A DPC-capable Downstream Port must support ERR_COR signaling, independent of whether it supports Advanced Error Reporting (AER) or not. DPC ERR_COR signaling is enabled by the DPC ERR_COR Enable bit in the DPC Control register. DPC triggering is indicated by the DPC

Trigger Status bit in the DPC Status register. DPC ERR_COR signaling is managed independently of DPC interrupts, and it is permitted to use both mechanisms concurrently.

If the DPC ERR_COR Enable bit is Set, and the Correctable Error Reporting Enable bit in the Device Control register is Set, the Port must send an ERR_COR Message each time the DPC Trigger Status bit transitions from Clear to Set. DPC ERR_COR signaling must not Set the Correctable Error Detected bit in the Device Status register, since this event is not handled as an error.

For a given DPC trigger event, if a Port is going to send both an ERR_COR Message and an MSI/MSI-X transaction, then the Port must send the ERR_COR Message prior to sending the MSI/MSI-X transaction. There is no corresponding requirement if the INTx mechanism is being used to signal DPC interrupts, since INTx Messages won't necessarily remain ordered with respect to ERR_COR Messages when passing through routing elements.

Formatted: Font color: Black



IMPLEMENTATION NOTE

Use of DPC ERR_COR Signaling

It is recommended that operating systems use DPC interrupts for signaling when DPC has been triggered. While DPC ERR_COR signaling indicates the same event, DPC ERR_COR signaling is primarily intended for use by platform firmware, when it needs to be notified in order to do its own logging of the event or provide "firmware first" services.

6.2.10.3. Root Port Programmed I/O (RP PIO) Error Controls

The RP PIO error control registers enable fine-grained control over what happens when Non-Posted Requests that are tracked by the Root Port encounter certain uncorrectable errors or Advisory Non-Fatal Errors. See Section 2.9.3 for a description of which Non-Posted Requests are tracked. A set of control and status bits exists for receiving Completion with Unsupported Request status (UR Cpl), receiving Completion with Completer Abort status (CA Cpl), and Completion Timeout (CTO) errors. Independent sets of these error bits exist for Configuration Requests, I/O Requests, and Memory Requests. This finer granularity enables more precise error handling for this subset of uncorrectable errors (UR Cpl, CA Cpl, and CTO). As a key example, UR Cpl errors with Memory Read Requests can be configured to trigger DPC for proper containment and error handling, while UR Cpl errors with Configuration Requests can be configured to return all 1's (without trigging DPC) for normal probing and enumeration.

A UR or CA error logged in AER is the result of the Root Port operating in the role of a Completer, and for a received Non-Posted Request, returning a Completion. In contrast, a UR Cpl or CA Cpl error logged as an RP PIO error is the result of the Root Port operating in the role of a Requester, and for an outstanding Non-Posted Request, receiving a Completion. CTO errors logged in both AER and RP PIO are the result of the Root Port operating in the role of a Requester, though the RP PIO error controls support per-space granularity. Depending upon the control register settings, CTO errors can be logged in AER registers, in RP PIO registers, or both. If software unmasks CTO errors in RP PIO, it is recommended that software mask CTO errors in AER in order to avoid unintended interactions.

The RP PIO Header Log, RP PIO ImpSpec Log, and RP PIO TLP Prefix Log registers are referred to collectively as the RP PIO log registers. The RP PIO Header Log must be implemented; the RP PIO ImpSpec Log and RP PIO TLP Prefix Log are optional. The RP PIO Log Size field indicates how many DWORDs are allocated for the RP PIO log registers, and from this the allocated size for the RP PIO TLP Prefix Log can be calculated. See Section 7.31.2.

The RP PIO Status, Mask, and Severity registers behave similarly to the Uncorrectable Error Status, Mask, and Severity registers in AER. See Sections 7.10.2, 7.10.3, and 7.10.4. When an RP PIO error is detected while it is unmasked, the associated bit in the RP PIO Status register is Set, and the error is recorded in the RP PIO log registers (assuming that RP PIO error logging resources are available). When an RP PIO error is detected while it is masked, the associated bit is still Set in the RP PIO Status register, but the error does not trigger DPC and the error is not recorded in the RP PIO log registers.

Each unmasked RP PIO error is handled either as an uncorrectable error or an Advisory Non-Fatal Error, as determined by the value of the corresponding bit in the RP PIO Severity register. If the associated Severity bit is Set, the error is handled as uncorrectable, triggering DPC (assuming that DPC is enabled) and signaling this event with a DPC interrupt and/or ERR_COR (if enabled). If the associated Severity bit is Clear, the error is handled as an Advisory Non-Fatal Error (without triggering DPC) and signaled with ERR_COR (if enabled).

Formatted: Font color: Auto



IMPLEMENTATION NOTE

Use of RP PIO Advisory Non-Fatal Errors

Each RP PIO error can be handled either as uncorrectable or an Advisory Non-Fatal error. Uncorrectable error handling usually logs the error, triggers DPC, and signals the event either with a DPC interrupt, an ERR_COR, or both. Advisory Non-Fatal Error handling usually logs the error and signals the event with ERR_COR.

RP PIO Advisory Non-Fatal Errors can be used by software in certain cases to handle RP PIO errors robustly without incurring the disruption caused if DPC is triggered in the RP. If an RP PIO Exception is not enabled for a given error, an all 1's value must be returned whenever the error occurs. If the error does not trigger DPC, software may be uncertain if the all 1's value returned by a given PIO read is the actual data value returned by the Completion versus indicating that an error occurred with that PIO read. If software enables Advisory Non-Fatal Error handling for that error, instances of that error will be logged, enabling software to distinguish the two cases.

The use of RP PIO Advisory Non-Fatal Errors is notably beneficial if DPC is triggered in a Switch Downstream Port, and that causes one or more Completion Timeouts in the RP as a side-effect, as described in Section 2.9.3. If the RP handles Completion Timeout errors as Advisory Non-Fatal, this avoids DPC being triggered in the RP, permitting continued operation with the other Switch Downstream Ports.

The RP PIO First Error Pointer, RP PIO Header Log, and RP PIO TLP Prefix Log behave similarly to the First Error Pointer, Header Log, and TLP Prefix Log in AER. The RP PIO First Error Pointer is defined to be valid when its value indicates a bit in the RP PIO Status register that is Set. When the RP PIO First Error Pointer is valid, the RP PIO log registers contain the information associated with the indicated error. The RP PIO ImpSpec Log, if implemented, contains implementation-specific information, e.g., the source of the Request TLP.

In contrast to AER, where the recording of CTO error information in the AER log registers is optional, RP PIO implementations must support recording RP PIO CTO error information in the RP PIO log registers.

The RP PIO SysError register provides a means to generate a System Error when an RP PIO error occurs. If an unmasked RP PIO error is detected while its associated bit in the RP PIO SysError register is Set, a System Error is generated.

The RP PIO Exception register provides a means to generate a synchronous processor exception[87] when an RP PIO error occurs with certain tracked Non-Posted Requests that are generated by a processor instruction. See Section 2.9.3. This exception must support all such tracked read Requests, and may optionally support Configuration write, I/O write, and AtomicOp Requests. If an RP PIO error with an exception-supported Non-Posted Request is detected while its associated bit in the RP PIO Exception register is Set, the processor instruction that generated the Non-Posted Request must take a synchronous exception. This is independent of whether the RP PIO error is masked or not.

The details of a processor instruction taking a synchronous exception are processor-specific, but at a minimum, the mechanism must be able to interrupt the normal processor instruction flow either before completion of the instruction that generated the Non-Posted Request, or immediately following that instruction. The intent is that exception handling routines in system firmware, the operating system, or both, can examine the cause of the exception and take corrective action if necessary.

If an RP PIO error occurs with a processor-generated read or AtomicOp Request, and the RP PIO Exception register value does not cause an exception, a value of all 1's must be returned for the instruction that generated the Request.

87 "Exception" is used as a generic term for a variety of mechanisms used by processors, including interrupts, traps, machine checks, instruction aborts, etc.

Formatted: Font color: Purple

IMPLEMENTATION NOTE

Synchronous Exception Implementation

The exact mechanism for implementing synchronous exceptions is processor and platform specific. One possible implementation is poisoning the data returned to the processor for a read or AtomicOp Request that encounters an error. While this approach is likely to work with those Requests, it might not work with Configuration and I/O write Requests since they return no data.

Another possible implementation is marking the response transaction for processor-generated Non-Posted Requests with some other type of indication of the Request having failed, e.g., a "hard fail" response. This approach is more likely to work with all processor-generated Non-Posted Requests.

Formatted: Font color: Purple

IMPLEMENTATION NOTE

Formatted: Std Title, Don't keep with next

RP PIO Mask Bit Behavior and Rationale

For a given RP PIO error, the associated mask bit in the RP PIO Mask register affects its associated status bit setting, error logging, and error signaling in a manner that closely parallels the behavior of mask bits in AER.

SysError generation for a given RP PIO error is primarily controlled by the associated bit in the RP PIO SysError register, but is also contingent upon the associated RP PIO mask bit being Clear. This behavior was chosen for consistency with AER, and also since it is poor practice to generate a SysError without logging the reason.

Exception generation for a given RP PIO error is independent of the associated RP PIO mask bit value. Usage Models are envisioned where an RP PIO error needs to generate an Exception without logging an RP PIO error or triggering DPC.

If a Root Port has outstanding tracked Non-Posted Requests when DPC is triggered, the Root Port is required to synthesize Completions for those tracked Requests, although this may be a logical synthesis, and not the actual creation of Completion TLPs. See Section 2.9.3. These synthesized Completions shall have a UR or CA Completion Status, as determined by the value of the DPC Completion Control bit. The UR/CA Completion Status, combined with the associated Request's targeted Space (Configuration, I/O, or Memory), then determines which RP PIO error control bits are used to govern the remainder of the error handling for each outstanding tracked Request. For example, a Root Port handling the case of a Configuration Read receiving a synthesized UR Completion would use the Cfg UR Cpl bit (bit 0) in the RP PIO SysError and RP PIO Exception registers to determine whether to generate a System Error, generate a processor instruction exception, or both.

Root Port error handling for tracked Non-Posted Requests with errors other than receiving UR and CA Completions is governed by a combination of AER and RP PIO error controls. Examples are CTO[88], Poisoned TLP Received, and Malformed TLP. For a given error managed by AER, the

[88] CTO errors have status and mask bits in both AER and RP PIO, though RP PIO has independent sets of bits for each of the 3 spaces. Other errors in AER have no equivalent errors in RP PIO.

associated AER Mask and Severity bits partially or completely determine if the error must be handled as an uncorrectable error, handled as an Advisory Non-Fatal Error, or handled as a masked error.

- If the AER-managed error is to be handled as an uncorrectable error (see Section 6.2.2.2), DPC is triggered. As described earlier, Completions are synthesized for any outstanding tracked Non-Posted Requests, and for each Request, the RP PIO SysError and RP PIO Exception bits associated with the Request type and Completion Status apply.
- If the AER-managed error is to be handled as an Advisory Non-Fatal Error (see Section 6.2.3.2.4) or a masked error (see Section 6.2.3.2.2), DPC is not triggered, so the RP PIO SysError and RP PIO Exception bits do not apply.
- One notable example is the case of a Root Port with an outstanding PIO read receiving an associated Completion that is poisoned. If the AER Severity bit is Set, forcing the error to be handled as an uncorrectable error, the RP PIO Exception bit determines if all 1's is returned versus generating an exception. If the error is to be handled as an Advisory Non-Fatal Error as described in Section 6.2.3.2.4.3, the poisoned data is propagated internally by the Root Port, and from there is handled in a platform specific manner. If the error is to be handled as a masked error, the poisoned data is handled in a platform specific manner.

6.2.10.4. Software Triggering of DPC

If the DPC Software Triggering Supported bit in the DPC Capability register is Set, then software can trigger DPC by writing a 1b to the DPC Software Trigger bit in the DPC Control register, assuming that DPC is enabled and the Port isn't currently in DPC. This mechanism is envisioned to be useful for software and/or firmware development and testing. It also supports usage models where software or firmware examines RP PIO Exceptions or RP PIO Advisory Non-Fatal Errors, and decides to trigger DPC based upon the situation.

When this mechanism triggers DPC, the DPC Trigger Reason and DPC Trigger Reason Extension fields in the DPC Status register will indicate this as the reason.

If a Port is already in DPC when a 1b is written to the DPC Software Trigger bit, the Port remains in DPC, and the DPC Trigger Reason and DPC Trigger Reason Extension fields are not modified.

IMPLEMENTATION NOTE

Avoid Disable Link and Hot-Plug Surprise Use With DPC

It is recommended that software not Set the Link Disable bit in the Link Control register while DPC is enabled but not triggered. Setting the Link Disable bit will cause the Link to be directed to DL_Down, invoking some semantics similar to those in DPC, but lacking others. The subsequent arrival of Posted Requests will likely trigger DPC soon, anyway.

Similarly, it is recommended that a Port that supports DPC not Set the Hot-Plug Surprise bit in the Slot Capabilities register. Having this bit Set blocks the reporting of Surprise Down errors, preventing DPC from being triggered by this important error, greatly reducing the benefit of DPC.

Formatted: Font color: Auto

6.2.10.5. DL_Active ERR_COR Signaling

Support for this feature is indicated by the DL_Active ERR_COR Signaling Supported bit in the DPC Capability register. The feature is enabled by the DL_ACTIVE ERR_COR Enable bit in the DPC Control register. The DL_ACTIVE state is indicated by the Data Link Layer Link Active bit in the Link Status register. DL_ACTIVE ERR_COR signaling is managed independently of Data Link Layer State Changed interrupts, and it is permitted to use both mechanisms concurrently.

If the DL_ACTIVE ERR_COR Enable bit is Set, and the Correctable Error Reporting Enable bit in the Device Control register is Set, the Port must send an ERR_COR Message each time the Link transitions into the DL_ACTIVE state. DL_ACTIVE ERR_COR signaling must not Set the Correctable Error Detected bit in the Device Status register, since this event is not handled as an error. In contrast to Data Link Layer State Changed interrupts, DL_Active ERR_COR signaling only indicates the Link enters the DL_Active state, not when the Link exits the DL_Active state.

For a given DL_ACTIVE event, if a Port is going to send both an ERR_COR Message and an MSI/MSI-X transaction, then the Port must send the ERR_COR Message prior to sending the MSI/MSI-X transaction. There is no corresponding requirement if the INTx mechanism is being used to signal DL_ACTIVE interrupts, since INTx Messages won't necessarily remain ordered with respect to ERR_COR Messages when passing through routing elements.

IMPLEMENTATION NOTE

Use of DL_ACTIVE ERR_COR Signaling

It is recommended that operating systems use Data Link Layer State Changed interrupts for signaling when DL_ACTIVE changes state. While DL_ACTIVE ERR_COR signaling indicates a subset of the same events, DL_ACTIVE ERR_COR signaling is primarily intended for use by platform firmware, when it needs to be notified in order to do Downstream Port configuration or provide "firmware first" services.

Formatted: Font color: Auto

6.3. Virtual Channel Support

6.3.1. Introduction and Scope

The Virtual Channel mechanism provides a foundation for supporting differentiated services within the PCI Express fabric. It enables deployment of independent physical resources that together with traffic labeling are required for optimized handling of differentiated traffic. Traffic labeling is supported using Traffic Class TLP-level labels. The policy for traffic differentiation is determined by the TC/VC mapping and by the VC-based, Port-based, and Function-based arbitration mechanisms. The TC/VC mapping depends on the platform application requirements. These requirements drive the choice of the arbitration algorithms and configurability/programmability of arbiters allows detailed tuning of the traffic servicing policy.

The definition of the Virtual Channel and associated Traffic Class mechanisms is covered in Chapter 2. The VC configuration/programming model is defined in Sections 7.11 and 7.18.

This section covers VC mechanisms from the system perspective. It addresses the next level of details on:

- Supported TC/VC configurations
- VC-based arbitration – algorithms and rules
- Traffic ordering considerations
- Isochronous support as a specific usage model

6.3.2. TC/VC Mapping and Example Usage

A Virtual Channel is established when one or more TCs are associated with a physical resource designated by a VC ID. Every Traffic Class that is supported on a given path within the fabric must be mapped to one of the enabled Virtual Channels. Every Port must support the default TC0/VC0 pair – this is "hardwired." Any additional TC mapping or additional VC resource enablement is optional and is controlled by system software using the programming model described in Sections 7.11 and 7.18.

The number of VC resources provisioned within a component or enabled within a given fabric may vary due to implementation and usage model requirements, due to Hot-Plug of disparate components with varying resource capabilities, or due to system software restricting what resources may be enabled on a given path within the fabric.

Some examples to illustrate:

- A set of components (Root Complex, Endpoints, Switches) may only support the mandatory VC0 resource that must have TC0 mapped to VC0. System software may, based on application usage requirements, map one or all non-zero TCs to VC0 as well on any or all paths within the fabric.
- A set of components may support two VC resources, e.g., VC0 and VC1. System software must map TC0/VC0 and in addition, may map one or all non-zero TC labels to either VC0 or VC1.

As above, these mappings may be enabled on any or all paths within the fabric. Refer to the examples below for additional information.

- A Switch may be implemented with eight Ports – seven x1 Links with two VC resources and one x16 Link with one VC resource. System software may enable both VC resources on the x1 Links and assign one or more additional TCs to either VC thus allowing the Switch to differentiate traffic flowing between any Ports. The x16 Link must also be configured to map any non-TC0 traffic to VC0 if such traffic is to flow on this Link. Note: multi-Port components (Switches and Root Complex) are required to support independent TC/VC mapping per Port.

In any of the above examples, system software has the ability to map one, all, or a subset of the TCs to a given VC. Should system software wish to restrict the number of traffic classes that may flow through a given Link, it may configure only a subset of the TCs to the enabled VC resources. Any TLP indicating a TC that has not been mapped to an enabled VC resource must be treated as a Malformed TLP. This is referred to as TC Filtering. Flow Control credits for this TLP will be lost, and an uncorrectable error will be generated, so software intervention will usually be required to restore proper operation after a TC Filtering event occurs.

A graphical example of TC filtering is illustrated in Figure 6-4, where TCs (2:6) are not mapped to the Link that connects Endpoint A and the Switch. This means that the TLPs with TCs (2:6) are not allowed between the Switch and Endpoint A.

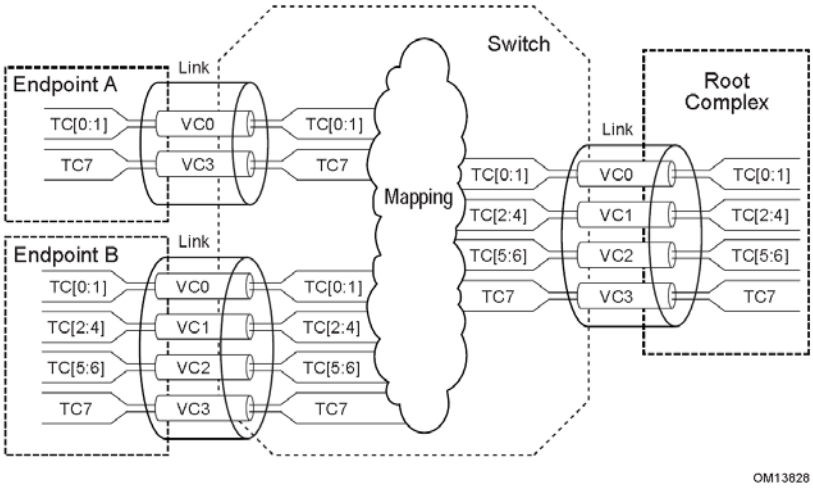


Figure 6-4: TC Filtering Example

Figure 6-5 shows an example of TC to VC mapping. A simple Switch with one Downstream Port and one Upstream Port connects an Endpoint to a Root Complex. At the Upstream Port, two VCs (VC0 and VC1) are enabled with the following mapping: TC(0-6)/VC0, TC7/VC1. At the Downstream Port, only VC0 is enabled and all TCs are mapped to VC0. In this example while TC7 is mapped to VC0 at the Downstream Port, it is re-mapped to VC1 at the Upstream Port. Although the Endpoint only supports VC0, when it labels transactions with different TCs, transactions

associated with TC7 from/to the Endpoint can take advantage of the second Virtual Channel enabled between the Switch and the Root Complex.

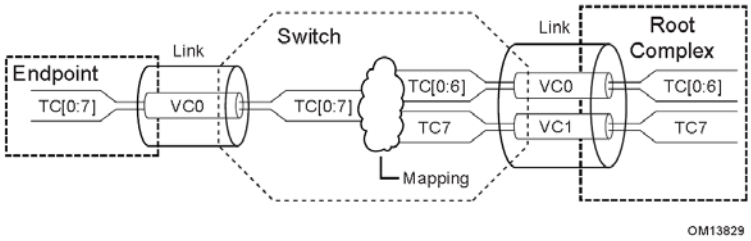


Figure 6-5: TC to VC Mapping Example

IMPLEMENTATION NOTE

Multiple TCs Over a Single VC

A single VC implementation may benefit from using multiple TCs. TCs provide ordering domains that may be used to differentiate traffic within the Endpoint or the Root Complex independent of the number of VCs supported.

In a simple configuration, where only VC0 is supported, traffic differentiation may not be accomplished in an optimum manner since the different TCs cannot be physically segregated. However, the benefits of carrying multiple TCs can still be exploited particularly in the small and "shallow" topologies where Endpoints are connected directly to Root Complex rather than through cascaded Switches. In these topologies traffic that is targeting Root Complex only needs to traverse a single Link, and an optimized scheduling of packets on both sides (Endpoint and Root Complex) based on TCs may accomplish significant improvement over the case when a single TC is used. Still, the inability to route differentiated traffic through separate resources with fully independent flow-control and independent ordering exposes all of the traffic to the potential head-of-line blocking conditions. Optimizing Endpoint internal architecture to minimize the exposure to the blocking conditions can reduce those risks.

6.3.3. VC Arbitration

Arbitration is one of the key aspects of the Virtual Channel mechanism and is defined in a manner that fully enables configurability to the specific application. In general, the definition of the VC-based arbitration mechanism is driven by the following objectives:

- To prevent false transaction timeouts and to guarantee data flow forward progress
- To provide differentiated services between data flows within the fabric
- To provide guaranteed bandwidth with deterministic (and reasonably small) end-to-end latency between components

Links are bidirectional, i.e., each Port can be an Ingress or an Egress Port depending on the direction of traffic flow. This is illustrated by the example of a 3-Port Switch in Figure 6-6, where the paths for traffic flowing between Switch Ports are highlighted with different types of lines. In

the following sections, VC Arbitration is defined using a Switch arbitration model since the Switch represents a functional superset from the arbitration perspective.

In addition, one-directional data flow is used in the description.

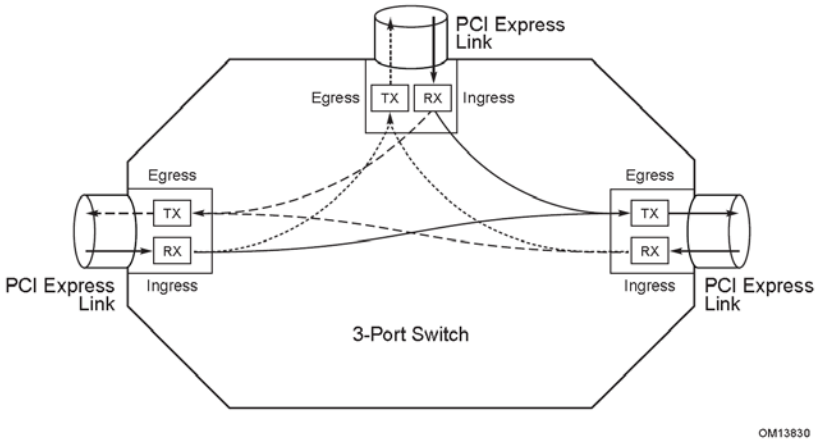


Figure 6-6: An Example of Traffic Flow Illustrating Ingress and Egress

6.3.3.1. Traffic Flow and Switch Arbitration Model

The following set of figures (Figure 6-7 and Figure 6-8) illustrates traffic flow through the Switch and summarizes the key aspects of the arbitration.

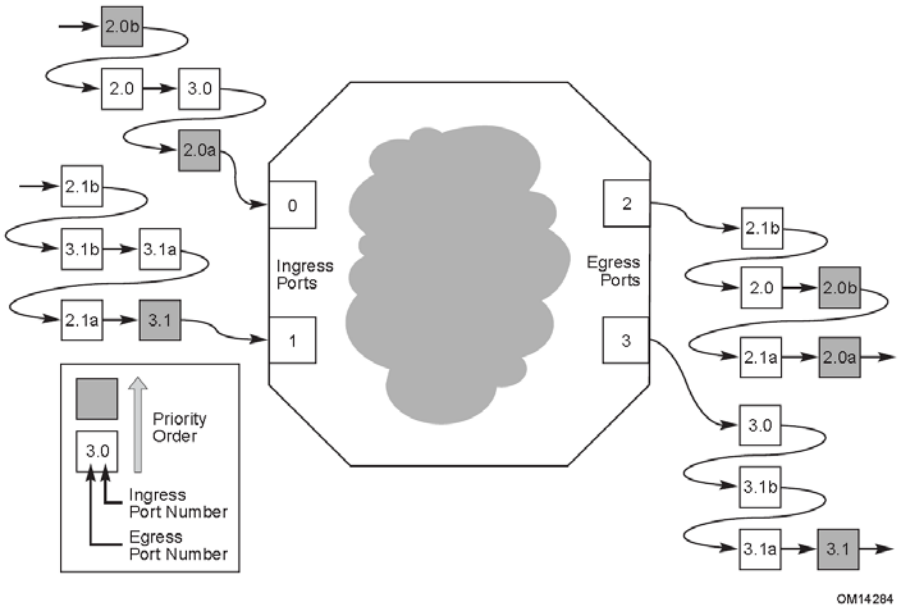


Figure 6-7: An Example of Differentiated Traffic Flow Through a Switch

At each Ingress Port an incoming traffic stream is represented in Figure 6-7 by small boxes. These boxes represent packets that are carried within different VCs that are distinguished using different levels of gray. Each of the boxes that represents a packet belonging to different VC includes designation of Ingress and Egress Ports to indicate where the packet is coming from and where it is going to. For example, designation "3.0" means that this packet is arriving at Port #0 (Ingress) and is destined to Port #3 (Egress). Within the Switch, packets are routed and serviced based on Switch internal arbitration mechanisms.

Switch arbitration model defines a required arbitration infrastructure and functionality within a Switch. This functionality is needed to support a set of arbitration policies that control traffic contention for an Egress Port from multiple Ingress Ports.

Figure 6-8 shows a conceptual model of a Switch highlighting resources and associated functionality in ingress to egress direction. Note that each Port in the Switch can have the role of an Ingress or Egress Port. Therefore, this figure only shows one particular scenario where the 4-Port Switch in this example has ingress traffic on Port #0 and Port #1, that targets Port #2 as an Egress Port. A different example may show different flow of traffic implying different roles for Ports on the Switch. The PCI Express architecture enables peer-to-peer communication through the Switch and, therefore, possible scenarios using the same example may include multiple separate and simultaneous ingress to egress flows (e.g., Port 0 to Port 2 and Port 1 to Port 3).

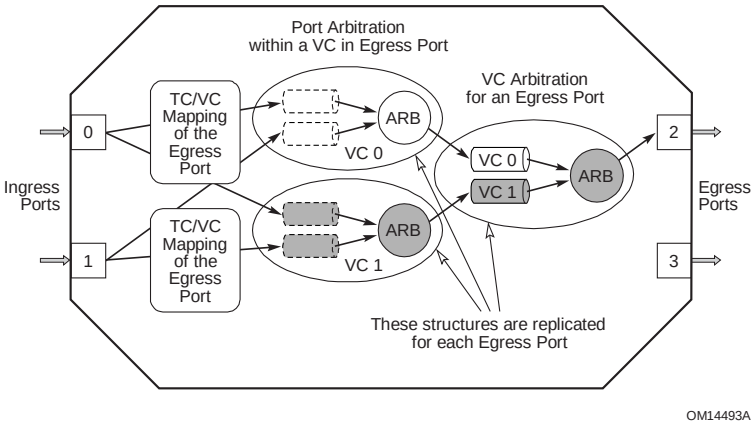


Figure 6-8: Switch Arbitration Structure

The following two steps conceptually describe routing of traffic received by the Switch on Port 0 and Port 1 and destined to Port 2. First, the target Egress Port is determined based on address/routing information in the TLP header. Secondly, the target VC of the Egress Port is determined based on the TC/VC map of the Egress Port. Transactions that target the same VC in the Egress Port but are from different Ingress Ports must be arbitrated before they can be forwarded to the corresponding resource in the Egress Port. This arbitration is referred to as the Port Arbitration.

Once the traffic reaches the destination VC resource in the Egress Port, it is subject to arbitration for the shared Link. From the Egress Port point of view this arbitration can be conceptually defined as a simple form of multiplexing where the multiplexing control is based on arbitration policies that are either fixed or configurable/programmable. This stage of arbitration between different VCs at an Egress Port is called the VC Arbitration of the Egress Port.

Independent of VC arbitration policy, a management/control logic associated with each VC must observe transaction ordering and flow control rules before it can make pending traffic visible to the arbitration mechanism.

IMPLEMENTATION NOTE

VC Control Logic at the Egress Port

VC control logic at every Egress Port includes:

- VC Flow Control logic
- VC Ordering Control logic

Flow control credits are exchanged between two Ports connected to the same Link. Availability of flow-control credits is one of the qualifiers that VC control logic must use to decide when a VC is allowed to compete for the shared Link resource (i.e., Data Link Layer transmit/retry buffer). If a candidate packet cannot be submitted due to the lack of an adequate number of flow control credits, VC control logic must mask the presence of pending packet to prevent blockage of traffic from other VCs. Note that since each VC includes buffering resources for Posted Requests, Non-Posted Requests, and Completion packets, the VC control logic must also take into account availability of flow control credits for the particular candidate packet. In addition, VC control logic must observe ordering rules (see Section 2.4 for more details) for Posted/Non-Posted/Completion transactions to prevent deadlocks and violation of producer-consumer ordering model.

6.3.3.2. VC Arbitration – Arbitration Between VCs

This specification defines a default VC prioritization via the VC Identification (VC ID) assignment, i.e., the VC IDs are arranged in ascending order of relative priority in the Virtual Channel Capability structure or Multi-Function Virtual Channel Capability structure. The example in Figure 6-9 illustrates a Port that supports eight VCs with VC0 treated as the lowest priority and VC7 as the highest priority.

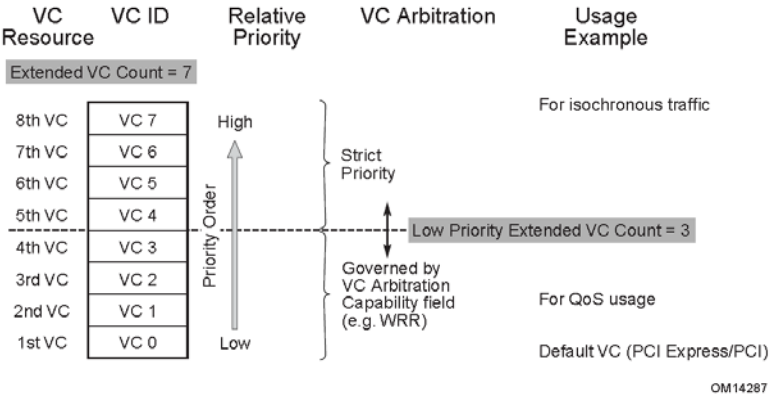


Figure 6-9: VC ID and Priority Order – An Example

The availability of default prioritization does not restrict the type of algorithms that may be implemented to support VC arbitration – either implementation-specific or one of the architecture-defined methods:

- Strict Priority – Based on inherent prioritization, i.e., VC0 = lowest, VC7 = highest
- Round Robin (RR) – Simplest form of arbitration where all VCs have equal priority
- Weighted RR – Programmable weight factor determines the level of service

If strict priority arbitration is supported by the hardware for a subset of the VC resources, software can configure the VCs into two priority groups – a lower and an upper group. The upper group is treated as a strict priority arbitration group while the lower group is arbitrated to only when there are no packets to process in the upper group. Figure 6-9 illustrates an example configuration that supports eight VCs separated into two groups – the lower group consisting of VC0-VC3 and the upper group consisting of VC4-VC7. The arbitration within the lower group can be configured to one of the supported arbitration methods. The Low Priority Extended VC Count field in the Port VC Capability register 1 indicates the size of this group. The arbitration methods are listed in the VC Arbitration Capability field in the Port VC Capability register 2. Refer to Sections 7.11 and 7.18 for details. When the Low Priority Extended VC Count field is set to zero, all VCs are governed by the strict-priority VC arbitration; when the field is equal to the Extended VC Count, all VCs are governed by the VC arbitration indicated by the VC Arbitration Capability field.

6.3.3.2.1. Strict Priority Arbitration Model

Strict priority arbitration enables minimal latency for high-priority transactions. However, there is potential danger of bandwidth starvation should it not be applied correctly. Using strict priority requires all high-priority traffic to be regulated in terms of maximum peak bandwidth and Link usage duration. Regulation must be applied either at the transaction injection Port/Function or within subsequent Egress Ports where data flows contend for a common Link. System software must configure traffic such that lower priority transactions will be serviced at a sufficient rate to avoid transaction timeouts.

6.3.3.2.2. Round Robin Arbitration Model

Round Robin arbitration is used to provide, at the transaction level, equal[89] opportunities to all traffic. Note that this scheme is used where different unordered streams need to be serviced with the same priority.

In the case where differentiation is required, a Weighted Round Robin scheme can be used. The WRR scheme is commonly used in the case where bandwidth regulation is not enforced by the sources of traffic and therefore it is not possible to use the priority scheme without risking starvation of lower priority traffic. The key is that this scheme provides fairness during traffic contention by allowing at least one arbitration win per arbitration loop. Assigned weights regulate both minimum allowed bandwidth and maximum burstiness for each VC during the contention. This means that it bounds the arbitration latency for traffic from different VCs. Note that latencies are also dependent on the maximum packet sizes allowed for traffic that is mapped onto those VCs.

One of the key usage models of the WRR scheme is support for QoS policy where different QoS levels can be provided using different weights.

[89] Note that this does not imply equivalence and fairness in the terms of bandwidth usage.

Although weights can be fixed (by hardware implementation) for certain applications, to provide more generic support for different applications, components that support the WRR scheme are recommended to implement programmable WRR. Programming of WRR is controlled using the software interface defined in Sections 7.11 and 7.18.

6.3.3.3. Port Arbitration – Arbitration Within VC

For Switches, Port Arbitration refers to the arbitration at an Egress Port between traffic coming from other Ingress Ports that is mapped to the same VC. For Root Ports, Port Arbitration refers to the arbitration at a Root Egress Port between peer-to-peer traffic coming from other Root Ingress Ports that is mapped to the same VC. For RCRBs, Port Arbitration refers to the arbitration at the RCRB (e.g., for host memory) between traffic coming from Root Ports that is mapped to the same VC. An inherent prioritization scheme for arbitration among VCs in this context is not applicable since it would imply strict arbitration priority for different Ports. Traffic from different Ports can be arbitrated using the following supported schemes:

- Hardware-fixed arbitration scheme, e.g., Round Robin
- Programmable WRR arbitration scheme
- Programmable Time-based WRR arbitration scheme

Hardware-fixed RR or RR-like scheme is the simplest to implement since it does not require any programmability. It makes all Ports equal priority, which is acceptable for applications where no software-managed differentiation or per-Port-based bandwidth budgeting is required.

Programmable WRR allows flexibility since it can operate as flat RR or if differentiation is required, different weights can be applied to traffic coming from different Ports in the similar manner as described in Section 6.3.3.2. This scheme is used where different allocation of bandwidth needs to be provided for different Ports.

A Time-based WRR is used for applications where not only different allocation of bandwidth is required but also a tight control of usage of that bandwidth. This scheme allows control of the amount of traffic that can be injected from different Ports within a certain fixed period of time. This is required for certain applications such as isochronous services, where traffic needs to meet a strict deadline requirement. Section 6.3.4 provides basic rules to support isochronous applications. For more details on time-based arbitration and on the isochronous service as a usage model for this arbitration scheme refer to Appendix A.

6.3.3.4. Multi-Function Devices and Function Arbitration

The multi-Function arbitration model defines an optional arbitration infrastructure and functionality within a multi-Function device. This functionality is needed to support a set of arbitration policies that control traffic contention for the device's Upstream Egress Port from its multiple Functions.

Figure 6-10 shows a conceptual model of a multi-Function device highlighting resources and associated functionality. Note that each Function optionally contains a VC Capability structure, which if present manages TC/VC mapping, optional Port Arbitration, and optional VC Arbitration, all within the Function. The MFVC Capability structure manages TC/VC mapping, optional Function Arbitration, and optional VC Arbitration for the device's Upstream Egress Port. Together these resources enable enhanced QoS management for Upstream requests. However, unlike a complete Switch with devices on its Downstream Ports, the multi-Function device model does not support full QoS management for peer-to-peer requests between Functions or for Downstream requests.

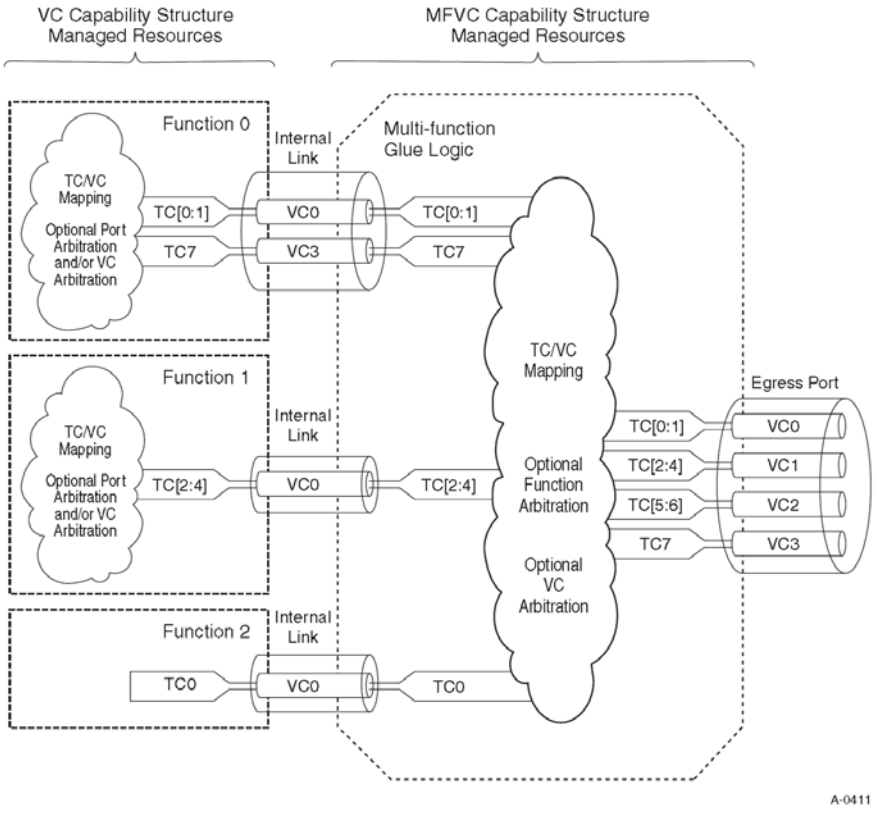


Figure 6-10: Multi-Function Arbitration Model

QoS for an Upstream request originating at a Function is managed as follows. First, a Function-specific mechanism applies a TC to the request. For example, a device driver might configure a Function to tag all its requests with TC7.

Next, if the Function contains a VC Capability structure, it specifies the TC/VC mapping to one of the Function's VC resources (perhaps the Function's single VC resource). In addition, the VC Capability structure supports the enablement and configuration of the Function's VC resources.

If the Function is a Switch and the target VC resource supports Port Arbitration, this mechanism governs how the Switch's multiple Downstream Ingress Ports arbitrate for that VC resource. If the Port Arbitration mechanism supports time-based WRR, this also governs the injection rate of requests from each Downstream Ingress Port.

If the Function supports VC arbitration, this mechanism manages how the Function's multiple VC resources arbitrate for the conceptual internal link to the MFVC resources.

Once a request packet conceptually arrives at MFVC resources, address/routing information in the TLP header determines whether the request goes Upstream or peer-to-peer to another Function. For the case of peer-to-peer, QoS management is left to unarchitected device-specific mechanisms. For the case of Upstream, TC/VC mapping in the MFVC Capability structure determines which VC resource the request will target. The MFVC Capability structure also supports enablement and configuration of the VC resources in the multi-Function glue logic. If the target VC resource supports Function Arbitration, this mechanism governs how the multiple Functions arbitrate for this VC resource. If the Function Arbitration mechanism supports time-based WRR, this governs the injection rate of requests for each Function into this VC resource.

Finally, if the MFVC Capability structure supports VC Arbitration, this mechanism governs how the MFVC's multiple VCs compete for the device's Upstream Egress Port. Independent of VC arbitration policy, management/control logic associated with each VC must observe transaction ordering and flow control rules before it can make pending traffic visible to the arbitration mechanism.



IMPLEMENTATION NOTE

Multi-Function Devices without the MFVC Capability Structure

If a multi-Function device lacks an MFVC Capability structure, the arbitration of data flows from different Functions of a multi-Function device is beyond the scope of this specification However, if a multi-Function device supports TCs other than TC0 and does not implement an MFVC Capability structure, it must implement a single VC Capability structure in Function 0 to provide architected TC/VC mappings for the Link.

6.3.4. Isochronous Support

Servicing isochronous data transfer requires a system to provide not only guaranteed data bandwidth but also deterministic service latency. The isochronous support mechanisms are defined to ensure that isochronous traffic receives its allocated bandwidth over a relevant period of time while also preventing starvation of the other traffic in the system. Isochronous support mechanisms apply to communication between Endpoint and Root Complex as well as to peer-to-peer communication.

Isochronous service is realized through proper use of mechanisms such as TC transaction labeling, VC data-transfer protocol, and TC-to-VC mapping. End-to-end isochronous service requires software to set up proper configuration along the path between the Requester and the Completer. This section describes the rules for software configuration and the rules hardware components must follow to provide end-to-end isochronous services. More information and background material regarding isochronous applications and isochronous service design guidelines can be found in Appendix A.

6.3.4.1. Rules for Software Configuration

System software must obey the following rules to configure PCI Express fabric for isochronous traffic:

- Software must designate one or more TCs for isochronous transactions.
- Software must ensure that the Attribute fields of all isochronous requests targeting the same Completer are fixed and identical.
- Software must configure all VC resources used to support isochronous traffic to be serviced (arbitrated) at the requisite bandwidth and latency to meet the application objectives. This may be accomplished using strict priority, WRR, or hardware-fixed arbitration.
- Software should not intermix isochronous traffic with non-isochronous traffic on a given VC.
- Software must observe the Maximum Time Slots capability reported by the Port or RCRB.
- Software must not assign all Link capacity to isochronous traffic. This is required to ensure the requisite forward progress of other non-isochronous transactions to avoid false transaction timeouts.
- Software must limit the Max_Payload_Size for each path that supports isochronous to meet the isochronous latency. For example, all traffic flowing on a path from an isochronous capable device to the Root Complex should be limited to packets that do not exceed the Max_Payload_Size required to meet the isochronous latency requirements.
- Software must set Max_Read_Request_Size of an isochronous-configured device with a value that does not exceed the Max_Payload_Size set for the device.

6.3.4.2. Rules for Requesters

A Requester requiring isochronous services must obey the following rules:

- The value in the Length field of read requests must never exceed Max_Payload_Size.
- If isochronous traffic targets the Root Complex and the RCRB indicates it cannot meet the isochronous bandwidth and latency requirements without requiring all transactions to set the No Snoop attribute bit, indicated by setting the Reject Snoop Transactions bit, then this bit must be set within the TLP header else the transaction will be rejected.

6.3.4.3. Rules for Completers

A Completer providing isochronous services must obey the following rules:

- A Completer should not apply flow control induced backpressure to uniformly injected isochronous requests under normal operating conditions.
- A Completer must report its isochronous bandwidth capability in the Maximum Time Slots field in the VC Resource Capability register. Note that a Completer must account for partial writes.
- A Completer must observe the maximum isochronous transaction latency.
- A Root Complex as a Completer must implement at least one RCRB and support time-based Port Arbitration for the associated VCs. Note that time-based Port Arbitration only applies to request transactions.

6.3.4.4. Rules for Switches and Root Complexes

A Switch providing isochronous services must obey the following rules. The same rules apply to Root Complexes that support isochronous data flows peer-to-peer between Root Ports, abbreviated in this section as "P2P-RC."

- An isochronous-configured Switch or P2P-RC Port should not apply flow control induced backpressure to uniformly injected isochronous requests under normal operating conditions.
- An isochronous-configured Switch or P2P-RC Port must observe the maximum isochronous transaction latency.
- A Switch or P2P-RC component must support time-based Port Arbitration for each Port that supports one or more VCs capable of supporting isochronous traffic. Note that time-based Port Arbitration applies to request transactions but not to completion transactions.

6.3.4.5. Rules for Multi-Function Devices

A multi-Function device that includes an MFVC Capability structure providing isochronous services must obey the following rules:

- MFVC glue logic configured for isochronous operation should not apply backpressure to uniformly injected isochronous requests from its Functions under normal operating conditions.
- The MFVC Capability structure must support time-based Function Arbitration for each VC capable of supporting isochronous traffic. Note that time-based Function Arbitration applies only to Upstream request transactions; it does not apply to any Downstream or peer-to-peer request transactions, nor to any completion transactions.

A multi-Function device that lacks an MFVC Capability structure has no architected mechanism to provide isochronous services for its multiple Functions concurrently.

6.4. Device Synchronization

System software requires a "stop" mechanism for ensuring that there are no outstanding transactions for a particular device in a system. For example, without such a mechanism renumbering Bus Numbers during system operation may cause the Requester ID (which includes the Bus Number) for a given device to change while Requests or Completions for that device are still in flight, and may thus be rendered invalid due to the change in the Requester ID. It is also desirable to be able to ensure that there are no outstanding transactions during a Hot-Plug orderly removal.

The details of stop mechanism implementation depend on the device hardware, device driver software, and system software. However, the fundamental requirements which must be supported to allow system software management of the fabric include the abilities to:

- Block the device from generating new Requests
- Block the generation of Requests issued to the device
- Determine that all Requests being serviced by the device have been completed
- Determine that all non-posted Requests initiated by the device have completed
- Determine that all posted Requests initiated by the device have reached their destination

The ability of the driver and/or system software to block new Requests from the device is supported by the Bus Master Enable, SERR# Enable, and Interrupt Disable bits in the Command register (Section 7.5.1.1) of each device Function, and other such control bits.

Requests issued to the device are generally under the direct control of the driver, so system software can block these Requests by directing the driver to stop generating them (the details of this communication are system software specific). Similarly, Requests serviced by the device are normally under the device driver's control, so determining the completion of such requests is usually trivial.

The Transactions Pending bit provides a consistent way on a per-Function basis for software to determine that all non-posted Requests issued by the device have been completed (see Section 7.8.5).

Determining that posted Requests have reached their destination is handled by generating a transaction to "flush" any outstanding Requests. Writes to system memory using TC0 will be flushed by host reads of the device, and so require no explicit flush protocol. Writes using TCs other than TC0 require some type of flush synchronization mechanism. The mechanism itself is implementation specific to the device and its driver software. However, in all cases the device hardware and software implementers should thoroughly understand the ordering rules described in Section 2.4. This is especially true if the Relaxed Ordering or ID-Based Ordering attributes are set for any Requests initiated by the device.



IMPLEMENTATION NOTE

Flush Mechanisms

In a simple case such as that of an Endpoint communicating only with host memory through TC0, "flush" can be implemented simply by reading from the Endpoint. If the Endpoint issues writes to main memory using TCs other than TC0, "flush" can be implemented with a memory read on the corresponding TCs directed to main memory. The memory read needs to be performed on all TCs that the Endpoint is using.

If a memory read is used to "flush" outstanding transactions, but no actual read is required, it may be desirable to use the zero-length read semantic described in Section 2.2.5.

Peer-to-peer interaction between devices requires an explicit synchronization protocol between the involved devices, even if all communication is through TC0. For a given system, the model for managing peer-to-peer interaction must be established. System software, and device hardware and software must then conform to this model. The requirements for blocking Request generation and determining completion of Requests match the requirements for non-peer interaction, however the determination that Posted Requests have reached peer destination device(s) requires an explicit synchronization mechanism. The mechanism itself is implementation specific to the device, its driver software, and the model used for the establishment and disestablishment of peer communications.

6.5. Locked Transactions

6.5.1. Introduction

Locked Transaction support is required to prevent deadlock in systems that use legacy software which causes the accesses to I/O devices. Note that some CPUs may generate locked accesses as a result of executing instructions that implicitly trigger lock. Some legacy software misuses these transactions and generates locked sequences even when exclusive access is not required. Since locked accesses to I/O devices introduce potential deadlocks apart from those mentioned above, as well as serious performance degradation, PCI Express Endpoints are prohibited from supporting locked accesses, and new software must not use instructions which will cause locked accesses to I/O devices. Legacy Endpoints support locked accesses only for compatibility with existing software.

Only the Root Complex is allowed to initiate Locked Requests on PCI Express. Locked Requests initiated by Endpoints and Bridges are not supported. This is consistent with limitations for locked transaction use outlined in the *PCI Local Bus Specification* (Appendix F- Exclusive Accesses).

This section specifies the rules associated with supporting locked accesses from the Host CPU to Legacy Endpoints, including the propagation of those transactions through Switches and PCI Express/PCI Bridges.

6.5.2. Initiation and Propagation of Locked Transactions - Rules

Locked transaction sequences are generated by the Host CPU(s) as one or more reads followed by a number of writes to the same location(s). When a lock is established, all other traffic is blocked from using the path between the Root Complex and the locked Legacy Endpoint or Bridge.

- A locked transaction sequence or attempted locked transaction sequence is initiated on PCI Express using the "lock"–type Read Request/Completion (MRdLk/CplDLk) and terminated with the Unlock Message
 - Locked Requests which are completed with a status other than Successful Completion do not establish lock (explained in detail in the following sections)
 - Regardless of the status of any of the Completions associated with a locked sequence, all locked sequences and attempted locked sequences must be terminated by the transmission of an Unlock Message.
 - MRdLk, CplDLk, and Unlock semantics are allowed only for the default Traffic Class (TC0)
 - Only one locked transaction sequence attempt may be in progress at a given time within a single hierarchy domain
- The Unlock Message is sent from the Root Complex down the locked transaction path to the Completer, and may be broadcast from the Root Complex to all Endpoints and Bridges
 - Any device which is not involved in the locked sequence must ignore this Message
- Any violation of the rules for initiation and propagation of locked transactions can result in undefined device and/or system behavior

The initiation and propagation of a locked transaction sequence through PCI Express is performed as follows:

- A locked transaction sequence is started with a MRdLk Request
 - Any successive reads for the locked transaction sequence must also use MRdLk Requests
 - The Completions for any MRdLk Request use the CplDLk Completion type for successful Requests, and the CPlLk Completion type for unsuccessful Requests
- If any read associated with a locked sequence is completed unsuccessfully, the Requester must assume that the atomicity of the lock is no longer assured, and that the path between the Requester and Completer is no longer locked
- All writes for the locked sequence use MWr Requests
- The Unlock Message is used to indicate the end of a locked sequence
 - A Switch propagates Unlock Messages to the locked Egress Port

- Upon receiving an Unlock Message, a Legacy Endpoint or Bridge must unlock itself if it is in a locked state
 - If not locked, or if the Receiver is a PCI Express Endpoint or Bridge which does not support lock, the Unlock Message is ignored and discarded

6.5.3. Switches and Lock - Rules

Switches must distinguish transactions associated with locked sequences from other transactions to prevent other transactions from interfering with the lock and potentially causing deadlock. The following rules cover how this is done. Note that locked accesses are limited to TC0, which is always mapped to VC0.

- When a Switch propagates a MRdLk Request from the Ingress Port (closest to the Root Complex) to the Egress Port, it must block all Requests which map to the default Virtual Channel (VC0) from being propagated to the Egress Port
 - If a subsequent MRdLk Request is Received at this Ingress Port addressing a different Egress Port, the behavior of the Switch is undefined

 Note: This sort of split-lock access is not supported by PCI Express and software must not cause such a locked access. System deadlock may result from such accesses.
- When the CplDLk for the first MRdLk Request is returned, if the Completion indicates a Successful Completion status, the Switch must block all Requests from all other Ports from being propagated to either of the Ports involved in the locked access, except for Requests which map to non-VC0 on the Egress Port
- The two Ports involved in the locked sequence must remain blocked as described above until the Switch receives the Unlock Message (at the Ingress Port for the initial MRdLk Request)
 - The Unlock Message must be forwarded to the locked Egress Port
 - The Unlock Message may be broadcast to all other Ports
 - The Ingress Port is unblocked once the Unlock Message arrives, and the Egress Port(s) which were blocked are unblocked following the Transmission of the Unlock Message out of the Egress Ports
 - Ports which were not involved in the locked access are unaffected by the Unlock Message

6.5.4. PCI Express/PCI Bridges and Lock - Rules

The requirements for PCI Express/PCI Bridges are similar to those for Switches, except that, because PCI Express/PCI Bridges use only the default Virtual Channel and Traffic Class, all other traffic is blocked during the locked access. The requirements on the PCI bus side of the PCI Express/PCI Bridge match the requirements for a PCI/PCI Bridge (see the *PCI-to-PCI Bridge Architecture Specification, Revision 1.2* and the *PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0*).

6.5.5. Root Complex and Lock - Rules

A Root Complex is permitted to support locked transactions as a Requester. If locked transactions are supported, a Root Complex must follow the sequence described in Section 6.5.2 to perform a locked access. The mechanisms used by the Root Complex to interface PCI Express to the Host CPU(s) are outside the scope of this document.

6.5.6. Legacy Endpoints

Legacy Endpoints are permitted to support locked accesses, although their use is discouraged. If locked accesses are supported, Legacy Endpoints must handle them as follows:

- The Legacy Endpoint becomes locked when it Transmits the first Completion for the first Read Request of the locked access with a Successful Completion status
 - If the completion status is not Successful Completion, the Legacy Endpoint does not become locked
 - Once locked, the Legacy Endpoint must remain locked until it receives the Unlock Message
- While locked, a Legacy Endpoint must not issue any Requests using TCs which map to the default Virtual Channel (VC0)

 Note that this requirement applies to all possible sources of Requests within the Endpoint, in the case where there is more than one possible source of Requests.
 - Requests may be issued using TCs which map to VCs other than the default Virtual Channel

6.5.7. PCI Express Endpoints

PCI Express Endpoints do not support lock. A PCI Express Endpoint must treat a MRdLk Request as an Unsupported Request (see Chapter 2).

6.6. PCI Express Reset - Rules

This section specifies the PCI Express Reset mechanisms. This section covers the relationship between the architectural mechanisms defined in this document and the reset mechanisms defined in this document. Any relationship between the PCI Express Conventional Reset and component or platform reset is component or platform specific (except as explicitly noted).

6.6.1. Conventional Reset

Conventional Reset includes all reset mechanisms other than Function Level Reset. There are two categories of Conventional Resets: Fundamental Reset and resets that are not Fundamental Reset. This section applies to all types of Conventional Reset.

In all form factors and system hardware configurations, there must, at some level, be a hardware mechanism for setting or returning all Port states to the initial conditions specified in this document

– this mechanism is called "Fundamental Reset." This mechanism can take the form of an auxiliary signal provided by the system to a component or adapter card, in which case the signal must be called PERST#, and must conform to the rules specified in Section 4.2.4.8.1. When PERST# is provided to a component or adapter, this signal must be used by the component or adapter as Fundamental Reset. When PERST# is not provided to a component or adapter, Fundamental Reset is generated autonomously by the component or adapter, and the details of how this is done are outside the scope of this document. If a Fundamental Reset is generated autonomously by the component or adapter, and if power is supplied by the platform to the component/adapter, the component/adapter must generate a Fundamental Reset to itself if the supplied power goes outside of the limits specified for the form factor or system.

- There are three distinct types of Conventional Reset: cold, warm, and hot:
 - A Fundamental Reset must occur following the application of power to the component. This is called a cold reset.
 - In some cases, it may be possible for the Fundamental Reset mechanism to be triggered by hardware without the removal and re-application of power to the component. This is called a warm reset. This document does not specify a means for generating a warm reset.
 - There is an in-band mechanism for propagating Conventional Reset across a Link. This is called a hot reset and is described in Section 4.2.4.8.2.

 There is an in-band mechanism for software to force a Link into Electrical Idle, "disabling" the Link. The Disabled LTSSM state is described in Section 4.2.5.9, ~~and~~ the Link Disable control bit is described in Section 7.8.7, and the Downstream Port Containment mechanism is described in Section 6.2.10. Disabling a Link causes Downstream components to undergo a hot reset.

 Note also that the Data Link Layer reporting DL_Down status is in some ways identical to a hot reset – see Section 2.9.
- On exit from any type of Conventional Reset (cold, warm, or hot), all Port registers and state machines must be set to their initialization values as specified in this document, except for sticky registers (see Section 7.4).
 - Note that, from a device point of view, any type of Conventional Reset (cold, warm, hot, or DL_Down) has the same effect at the Transaction Layer and above as would RST# assertion and de-assertion in conventional PCI.
- On exit from a Fundamental Reset, the Physical Layer will attempt to bring up the Link (see Section 4.2.5). Once both components on a Link have entered the initial Link Training state, they will proceed through Link initialization for the Physical Layer and then through Flow Control initialization for VC0, making the Data Link and Transaction Layers ready to use the Link.
 - Following Flow Control initialization for VC0, it is possible for TLPs and DLLPs to be transferred across the Link.

Following a Conventional Reset, some devices may require additional time before they are able to respond to Requests they receive. Particularly for Configuration Requests it is necessary that components and devices behave in a deterministic way, which the following rules address.

Comment [VS205]: ECN: Downstream Port containment (DPC)

The first set of rules addresses requirements for components and devices:

- A component must enter the LTSSM Detect state within 20 ms of the end of Fundamental Reset (Link Training is described in Section 4.2.4).
 - Note: In some systems, it is possible that the two components on a Link may exit Fundamental Reset at different times. Each component must observe the requirement to enter the initial active Link Training state within 20 ms of the end of Fundamental Reset from its own point of view.
- On the completion of Link Training (entering the DL_Active state, see Section 3.2), a component must be able to receive and process TLPs and DLLPs.

The second set of rules addresses requirements placed on the system:

- To allow components to perform internal initialization, system software must wait a specified minimum period following the end of a Conventional Reset of one or more devices before it is permitted to issue Configuration Requests to those devices, unless Readiness Notifications mechanisms are used (see Section 6.23.).
 - With a Downstream Port that does not support Link speeds greater than 5.0 GT/s, software must wait a minimum of 100 ms before sending a Configuration Request to the device immediately below that Port.
 - With a Downstream Port that supports Link speeds greater than 5.0 GT/s, software must wait a minimum of 100 ms after Link training completes before sending a Configuration Request to the device immediately below that Port. Software can determine when Link training completes by polling the Data Link Layer Link Active bit or by setting up an associated interrupt (see Section 6.7.3.3).
 - A system must guarantee that all components intended to be software visible at boot time are ready to receive Configuration Requests within the applicable minimum period based on the end of Conventional Reset at the Root Complex – how this is done is beyond the scope of this specification.
 - Note: Software should use 100 ms wait periods only if software enables CRS Software Visibility. Otherwise, Completion timeouts, platform timeouts, or lengthy processor instruction stalls may result. See the Configuration Request Retry Status Implementation Note in Section 2.3.1.
- Unless Readiness Notifications mechanisms are used (see Section 6.23), t~~T~~he Root Complex and/or system software must allow at least 1.0 s after a Conventional Reset of a device, before it may determine that a device which fails to return a Successful Completion status for a valid Configuration Request is a broken device. This period is independent of how quickly Link training completes.

 Note: This delay is analogous to the T_{rhfa} parameter specified for PCI/PCI-X, and is intended to allow an adequate amount of time for devices which require self initialization.
- When attempting a Configuration access to devices on a PCI or PCI-X bus segment behind a PCI Express/PCI(-X) Bridge, the timing parameter T_{rhfa} must be respected.

For this second set of rules, if system software does not have direct visibility into the state of Fundamental Reset (e.g., Hot-Plug; see Section 6.7), software must base these timing parameters on an event known to occur after the end of Fundamental Reset.

Comment [VS206]: D-ECN-Readiness Notifications

Comment [VS207]: D-ECN-Readiness Notifications

When a Link is in normal operation, the following rules apply:

- If, for whatever reason, a normally operating Link goes down, the Transaction and Data Link Layers will enter the DL_Inactive state (see Sections 2.9 and 3.2.1).
- For any Root or Switch Downstream Port, setting the Secondary Bus Reset bit of the Bridge Control register associated with the Port must cause a hot reset to be sent (see Section 4.2.4.8.2).
- For a Switch, the following must cause a hot reset to be sent on all Downstream Ports:
 - Setting the Secondary Bus Reset bit of the Bridge Control register associated with the Upstream Port
 - The Data Link Layer of the Upstream Port reporting DL_Down status. In Switches that support Link speeds greater than 5.0 GT/s, the Upstream Port must direct the LTSSM of each Downstream Port to the Hot Reset state, but not hold the LTSSMs in that state. This permits each Downstream Port to begin Link training immediately after its hot reset completes. This behavior is recommended for all Switches.
 - Receiving a hot reset on the Upstream Port

Certain aspects of Fundamental Reset are specified in this document and others are specific to a platform, form factor and/or implementation. Specific platforms, form factors or application spaces may require the additional specification of the timing and/or sequencing relationships between the components of the system for Fundamental Reset. For example, it might be required that all PCI Express components within a chassis observe the assertion and deassertion of Fundamental Reset at the same time (to within some tolerance). In a multi-chassis environment, it might be necessary to specify that the chassis containing the Root Complex be the last to exit Fundamental Reset.

In all cases where power and PERST# are supplied, the following parameters must be defined:

- T_{pvperl} – PERST# must remain active at least this long after power becomes valid
- T_{perst} – When asserted, PERST# must remain asserted at least this long
- T_{fail} – When power becomes invalid, PERST# must be asserted within this time

Additional parameters may be specified.

In all cases where a reference clock is supplied, the following parameter must be defined:

- $T_{perst\text{-}clk}$ – PERST# must remain active at least this long after any supplied reference clock is stable

Additional parameters may be specified.

6.6.2. Function-Level Reset (FLR)

The FLR mechanism enables software to quiesce and reset Endpoint hardware with Function-level granularity. Three example usage models illustrate the benefits of this feature:

- In some systems, it is possible that the software entity that controls a Function will cease to operate normally. To prevent data corruption, it is necessary to stop all PCI Express and

external I/O (not PCI Express) operations being performed by the Function. Other defined reset operations do not guarantee that external I/O operations will be stopped.

- In a partitioned environment where hardware is migrated from one partition to another, it is necessary to ensure that no residual "knowledge" of the prior partition be retained by hardware, for example, a user's secret information entrusted to the first partition but not to the second. Further, due to the wide range of Functions, it is necessary that this be done in a Function-independent way.
- When system software is taking down the software stack for a Function and then rebuilding that stack, it is sometimes necessary to return the state to an uninitialized state before rebuilding the Function's software stack.

Implementation of FLR is optional (not required), but is strongly recommended.

FLR applies on a per Function basis. Only the targeted Function is affected by the FLR operation. The Link state must not be affected by an FLR.

FLR modifies the Function state described by this specification as follows:

- Function registers and Function-specific state machines must be set to their initialization values as specified in this document, except for the following:
 - sticky-type registers (ROS, RWS, RW1CS)
 - registers defined as type HwInit
 - these other fields or registers:
 - Captured Slot Power Limit Value in the Device Capabilities register
 - Captured Slot Power Limit Scale in the Device Capabilities register
 - Max_Payload_Size in the Device Control register
 - Active State Power Management (ASPM) Control in the Link Control register
 - Read Completion Boundary (RCB) in the Link Control register
 - Common Clock Configuration in the Link Control register
 - Extended Synch in the Link Control register
 - Enable Clock Power Management in the Link Control register
 - Hardware Autonomous Width Disable in Link Control register
 - Hardware Autonomous Speed Disable in the Link Control 2 register
 - Link Equalization Request in the Link Status 2 register
 - Link Equalization Control register in the Secondary PCI Express Extended Capability structure
 - All registers in the Virtual Channel Capability structure
 - All registers in the Multi-Function Virtual Channel Capability structure

Note that the controls that enable the Function to initiate requests on PCI Express are cleared, including Bus Master Enable, MSI Enable, and the like, effectively causing the Function to become

quiescent on the Link.

Note that Port state machines associated with Link functionality including those in the Physical and Data Link Layers are not reset by FLR, and VC0 remains initialized following an FLR.

- Any outstanding INTx interrupt asserted by the Function must be de-asserted by sending the corresponding Deassert_INTx Message prior to starting the FLR.

 Note that when the FLR is initiated to a Function of a multi-Function device, if another Function continues to assert a matching INTx, no Deassert_INTx Message will be transmitted.

After an FLR has been initiated by writing a 1b to the Initiate Function Level Reset bit, the Function must complete the FLR within 100 ms. If software initiates an FLR when the Transactions Pending bit is 1b, then software must not initialize the Function until allowing adequate time for any associated Completions to arrive, or to achieve reasonable certainty that any remaining Completions will never arrive. For this purpose, it is recommended that software allow as much time as provided by the pre-FLR value for Completion Timeout on the device. If Completion Timeouts were disabled on the Function when FLR was issued, then the delay is system dependent but must be no less than 100 ms. If Function Readiness Status (FRS – see Section 6.23.2) is implemented, then system software is permitted to issue Configuration Requests to the Function immediately following receipt of an FRS Message indicating Configuration-Ready, however, this does not necessarily indicate that outstanding Requests initiated by the Function have completed.

Comment [VS208]: RC_ECN_RN_27_Aug_2013

Note that upon receipt of an FLR, a device Function may either clear all transaction status including Transactions Pending or set the Completion Timeout to its default value so that all pending transactions will time out during FLR execution. Regardless, the Transactions Pending bit must be clear upon completion of the FLR.

Since FLR modifies Function state not described by this specification (in addition to state that is described by this specification), it is necessary to specify the behavior of FLR using a set of criteria that, when applied to the Function, show that the Function has satisfied the requirements of FLR. The following criteria must be applied using Function-specific knowledge to evaluate the Function's behavior in response to an FLR:

- The Function must not give the appearance of an initialized adapter with an active host on any external interfaces controlled by that Function. The steps needed to terminate activity on external interfaces are outside of the scope of this specification.
 - For example, a network adapter must not respond to queries that would require adapter initialization by the host system or interaction with an active host system, but is permitted to perform actions that it is designed to perform without requiring host initialization or interaction. If the network adapter includes multiple Functions that operate on the same external network interface, this rule affects only those aspects associated with the particular Function reset by FLR.
- The Function must not retain within itself software readable state that potentially includes secret information associated with any preceding use of the Function. Main host memory assigned to the Function must not be modified by the Function.
 - For example, a Function with internal memory readable directly or indirectly by host software must clear or randomize that memory.

- The Function must return to a state such that normal configuration of the Function's PCI Express interface will cause it to be useable by drivers normally associated with the Function

When an FLR is initiated, the targeted Function must behave as follows:

- The Function must return the Completion for the configuration write that initiated the FLR operation and then initiate the FLR.
- While an FLR is in progress:
 - If a Request arrives, the Request is permitted to be silently discarded (following update of flow control credits) without logging or signaling it as an error.
 - If a Completion arrives, the Completion is permitted to be handled as an Unexpected Completion or to be silently discarded (following update of flow control credits) without logging or signaling it as an error.
 - While a Function is required to complete the FLR operation within the time limit described above, the subsequent Function-specific initialization sequence may require additional time. If additional time is required, the Function must return a Configuration Request Retry Status (CRS) Completion Status when a Configuration Request is received after the time limit above. After the Function responds to a Configuration Request with a Completion status other than CRS, it is not permitted to return CRS until it is reset again.



IMPLEMENTATION NOTE

Avoiding Data Corruption From Stale Completions

An FLR causes a Function to lose track of any outstanding non-posted Requests. Any corresponding Completions that later arrive are referred to as being "stale". If software issues an FLR while there are outstanding Requests, and then re-enables the Function for operation without waiting for potential stale Completions, any stale Completions that arrive afterwards may cause data corruption by being mistaken by the Function as belonging to Requests issued since the FLR.

Software can avoid data corruption from stale Completions in a variety of ways. Here's a possible algorithm:

1. Software that's performing the FLR synchronizes with other software that might potentially access the Function directly, and ensures such accesses do not occur during this algorithm.
2. Software clears the entire Command register, disabling the Function from issuing any new Requests.
3. Software polls the Transactions Pending bit in the Device Status register either until it is clear or until it has been long enough that software is reasonably certain that Completions associated with any remaining outstanding Transactions will never arrive. On many platforms, the Transactions Pending bit will usually clear within a few milliseconds, so software might choose to poll during this initial period using a tight software loop. On rare cases when the Transactions Pending bit does not clear by this time, software will need to poll for a much longer platform-specific period (potentially seconds), so software might choose to conduct this polling using a timer-based interrupt polling mechanism.
4. Software initiates the FLR.
5. Software waits 100 ms.
6. Software reconfigures the Function and enables it for normal operation.

6.7. PCI Express Hot-Plug Support

The PCI Express architecture is designed to natively support both hot-add and hot-removal ("hot-plug") of ~~adapters~~ cables, add-in cards, and modules. PCI Express hot-plug support ~~and~~ provides a "toolbox" of mechanisms that allow different user/operator models to be supported using a self-consistent infrastructure. These mechanisms may be used to implement orderly addition/removal that relies on coordination with the operating system (e.g., traditional PCI hot-plug), as well as async removal which proceeds without lock-step synchronization with the operating system.This section defines the set of hot-plug mechanisms and specifies how the elements of hot-plug, such as indicators and push buttons, must behave if implemented in a system.

Comment [VS209]: ECN: Downstream Port containment (DPC)

6.7.1. Elements of Hot-Plug

Table 6-6 lists the physical elements comprehended in this specification for support of hot-plug models. A form factor specification must define how these elements are used in that form factor. For a given form factor specification, it is possible that only some of the available hot-plug elements are required, or even that none of these elements are required. In all cases, the form factor specification must define all assumptions and limitations placed on the system or the user by the choice of elements included. Silicon component implementations that are intended to be used only with selected form factors are permitted to support only those elements that are required by the associated form factor(s).

Table 6-6: Elements of Hot-Plug

Element	Purpose
Indicators	Show the power and attention state of the slot
Manually-operated Retention Latch (MRL)	Holds adapter in place
MRL Sensor	Allows the Port and system software to detect the MRL being opened
Electromechanical Interlock	Prevents removal of adapter from slot
Attention Button	Allows user to request hot-plug operations
Software User Interface	Allows user to request hot-plug operations
Slot Numbering	Provides visual identification of slots
Power Controller	Software-controlled electronic component or components that control power to a slot or adapter and monitor that power for fault conditions
Out-of-band Presence Detect	Method of determining physical presence of an adapter in a slot that does not rely on the Physical Layer

Comment [VS210]: ECN: Downstream Port containment (DPC)

6.7.1.1. Indicators

Two indicators are defined: the Power Indicator and the Attention Indicator. Each indicator is in one of three states: on, off, or blinking. Hot-plug system software has exclusive control of the indicator states by writing the command registers associated with the indicator (with one exception noted below). The indicator requirements must be included in all form factor specifications. For a given form factor, the indicators may be required or optional or not applicable at all.

The hot-plug capable Port controls blink frequency, duty cycle, and phase of the indicators. Blinking indicators must operate at a frequency of between 1 and 2 Hz, with a 50% (+/- 5%) duty cycle. Blinking indicators are not required to be synchronous or in-phase between Ports.

Indicators may be physically located on the chassis or on the adapter (see the associated form factor specification for Indicator location requirements). Regardless of the physical location, logical control of the indicators is by the Downstream Port of the Upstream component on the Link.

The Downstream Port must not change the state of an indicator unless commanded to do so by software, except for platforms capable of detecting stuck-on power faults (relevant only when a power controller is implemented). In the case of a stuck-on power fault, the platform is permitted to override the Downstream Port and force the Power Indicator to be on (as an indication that the adapter should not be removed). The handling by system software of stuck-on faults is optional and not described in this specification. Therefore, the platform vendor must ensure that this feature, if implemented, is addressed via other software, platform documentation, or by other means.

6.7.1.1.1. Attention Indicator

The Attention Indicator, which must be yellow or amber in color, indicates that an operational problem exists or that the hot-plug slot is being identified so that a human operator can locate it easily.

Table 6-7: Attention Indicator States

Indicator Appearance	Meaning
Off	Normal - Normal operation
On	Attention - Operational problem at this slot
Blinking	Locate - Slot is being identified at the user's request

Attention Indicator Off

The Attention Indicator in the Off state indicates that neither the adapter (if one is present) nor the hot-plug slot requires attention.

Attention Indicator On

The Attention Indicator in the On state indicates that an operational problem exists at the adapter or slot.

An operational problem is a condition that prevents continued operation of an adapter. The operating system or other system software determines whether a specific condition prevents continued operation of an adapter and whether lighting the Attention Indicator is appropriate.

Examples of operational problems include problems related to external cabling, adapter, software drivers, and power faults. In general, the Attention Indicator in the On state indicates that an operation was attempted and failed or that an unexpected event occurred.

The Attention Indicator is not used to report problems detected while validating the request for a hot-plug operation. Validation is a term applied to any check that system software performs to assure that the requested operation is viable, permitted, and will not cause problems. Examples of validation failures include denial of permission to perform a hot-plug operation, insufficient power budget, and other conditions that may be detected before a hot-plug request is accepted.

Attention Indicator Blinking

A blinking Attention Indicator indicates that system software is identifying this slot for a human operator to find. This behavior is controlled by a user (for example, from a software user interface or management tool).

6.7.1.1.2. Power Indicator

The Power Indicator, which must be green in color, indicates the power state of the slot. Table 6-8 lists the Power Indicator states.

Table 6-8: Power Indicator States

Indicator Appearance	Meaning
Off	Power Off - Insertion or removal of the adapter is permitted.
On	Power On - Insertion or removal of the adapter is not permitted.
Blinking	Power Transition - Hot-plug operation is in progress and insertion or removal of the adapter is not permitted.

Power Indicator Off

The Power Indicator in the Off state indicates that insertion or removal of an the adapter is permitted. Main power to the slot is off if required by the form factor. Note that, depending on the form factor, other power/signals may remain on, even when main power is off and the Power Indicator is off. In an example using the PCI Express card form factor, if the platform provides Vaux to hot-plug slots and the MRL is closed, any signals switched by the MRL are connected to the slot even when the Power Indicator is off. Signals switched by the MRL are disconnected when the MRL is opened. System software must cause a slot's Power Indicator to be turned off when the slot is not powered and/or it is permissible to insert or remove an adapter. Refer to the appropriate form factor specification for details.

Power Indicator On

The Power Indicator in the On state indicates that the hot-plug operation is complete and that main power to the slot is On and that insertion or removal of the adapter is not permitted.

Power Indicator Blinking

A blinking Power Indicator indicates that the slot is powering up or powering down and that insertion or removal of the adapter is not permitted.

The blinking Power Indicator also provides visual feedback to the operator when the Attention Button is pressed or when hot-plug operation is initiated through the hot-plug software interface.

6.7.1.2. Manually-operated Retention Latch (MRL)

An MRL is a manually-operated retention mechanism that holds an adapter in the slot and prevents the user from removing the device. The MRL rigidly holds the adapter in the slot so that cables may be attached without the risk of creating intermittent contact. MRLs that hold down two or more adapters simultaneously are permitted in platforms that do not provide MRL Sensors.

6.7.1.3. MRL Sensor

The MRL Sensor is a switch, optical device, or other type of sensor that reports the position of a slot's MRL to the Downstream Port. The MRL Sensor reports closed when the MRL is fully closed and open at all other times (that is, if the MRL fully open or in an intermediate position).

If a power controller is implemented for the slot, the slot main power must be automatically removed from the slot when the MRL Sensor indicates that the MRL is open. If signals such as Vaux and SMBus are switched by the MRL, then these signals must be automatically removed from the slot when the MRL Sensor indicates that the MRL is open and must be restored to the slot when the MRL Sensor indicates that MRL has closed again. Refer to the appropriate form factor specification to identify the signals, if any, switched by the MRL.

Note that the Hot-Plug Controller does not autonomously change the state of either the Power Indicator or the Attention Indicator based on MRL sensor changes.



IMPLEMENTATION NOTE

MRL Sensor Handling

In the absence of an MRL sensor, for some form factors, out-of-band presence detect ~~staggered presence detect pins~~ may be used to handle the switched signals. In this case, when out-of-band presence detect indicates the absence of an adapter in a slot~~the presence pins break contact~~, the switched signals will be automatically removed from the slot.

If an MRL Sensor is implemented without a corresponding MRL Sensor input on the Hot-Plug Controller, it is recommended that the MRL Sensor be routed to power fault input of the Hot-Plug Controller. This allows an active adapter to be powered off when the MRL is opened.

Comment [VS211]: ECN: Downstream Port containment (DPC)

6.7.1.4. Electromechanical Interlock

An electromechanical interlock is a mechanism for physically locking the adapter or MRL in place until system software releases it. The state of the electromechanical interlock is set by software and must not change except in response to a subsequent software command. In particular, the state of the electromechanical interlock must be maintained even when power to the hot-plug slot is removed.

The current state of the electromechanical interlock must be reflected at all times in the Electromechanical Interlock Status bit in the Slot Status register, which must be updated within 200 ms of any commanded change. Software must wait at least 1 second after issuing a command to toggle the state of the Electromechanical Interlock before another command to toggle the state can be issued. Systems may optionally expand control of interlocks to provide physical security of the adapter.

6.7.1.5. Attention Button

The Attention Button is a momentary-contact push button switch, located adjacent to each hot-plug slot or on the adapter that is pressed by the user to initiate a hot-plug operation at that slot. Regardless of the physical location of the button, the signal is processed and indicated to software by hot-plug hardware associated with the Downstream Port corresponding to the slot.

The Attention Button must allow the user to initiate both hot add and hot remove operations regardless of the physical location of the button.

If present, the Power Indicator provides visual feedback to the human operator (if the system software accepts the request initiated by the Attention Button) by blinking. Once the Power Indicator begins blinking, a 5-second abort interval exists during which a second depression of the Attention Button cancels the operation.

If an operation initiated by an Attention Button fails for any reason, it is recommended that system software present an error message explaining the failure via a software user interface or add the error message to a system log.

6.7.1.6. Software User Interface

System software provides a user interface that allows hot insertions and hot removals to be initiated and that allows occupied slots to be monitored. A detailed discussion of hot-plug user interfaces is operating system specific and is therefore beyond the scope of this document.

On systems with multiple hot-plug slots, the system software must allow the user to initiate operations at each slot independent of the states of all other slots. Therefore, the user is permitted to initiate a hot-plug operation on one slot using either the software user interface or the Attention Button while a hot-plug operation on another slot is in process, regardless of which interface was used to start the first operation.

6.7.1.7. Slot Numbering

A Physical Slot Identifier (as defined in *PCI Hot-Plug Specification, Revision 1.1*, Section 1.5) consists of an optional chassis number and the physical slot number of the slot. The physical slot number is a

chassis unique identifier for a slot. System software determines the physical slot number from registers in the Port. Chassis number 0 is reserved for the main chassis. The chassis number for other chassis must be a non-zero value obtained from a PCI-to-PCI Bridge's Chassis Number register (see the *PCI-to-PCI Bridge Architecture Specification, Revision 1.2*, Section 13.4).

Regardless of the form factor associated with each slot, each physical slot number must be unique within a chassis.

6.7.1.8. Power Controller

The power controller is an element composed of one or more discrete components that acts under control of software to set the power state of the hot-plug slot as appropriate for the specific form factor. The power controller must also monitor the slot for power fault conditions (as defined in the associated form factor specification) that occur on the slot's main power rails and, if supported, auxiliary power rail.

If a power controller is not present, the power state of the hot-plug slot must be set automatically by the hot-plug controller in response to changes in the presence of an adapter in the slot.

The power controller monitors main and auxiliary power faults independently. If a power controller detects a main power fault on the hot-plug slot, it must automatically set its internal main power fault latch and remove main power from the hot-plug slot (without affecting auxiliary power). Similarly, if a power controller detects an auxiliary power fault on the hot-plug slot, it must automatically set its internal auxiliary power fault latch and remove auxiliary power from the hot-plug slot (without affecting main power). Power must remain off to the slot as long as the power fault condition remains latched, regardless of any writes by software to turn on power to the hot-plug slot. The main power fault latch is cleared when software turns off power to the hot-plug slot. The mechanism by which the auxiliary power fault latch is cleared is form factor specific but generally requires auxiliary power to be removed from the hot-plug slot. For example, one form factor may remove auxiliary power when the MRL for the slot is opened while another may require the adapter to be physically removed from the slot. Refer to the associated form factor specifications for specific requirements.

Since the Power Controller Control bit in the Slot Control register reflects the last value written and not the actual state of the power controller, this means there may be an inconsistency between the value of the Power Controller Control bit and the state of the power to the slot in a power fault condition. To determine whether slot is off due to a power fault, software must use the power fault software notification to detect power faults. To determine that a requested power-up operation has otherwise failed, software must use the hot-plug slot power-up time out mechanism described in Section 6.7.3.3.

Software must not assume that writing to the Slot Control register to change the power state of a hot-plug slot causes an immediate power state transition. After turning power on, software must wait for a Data Link Layer State Changed event, as described in Section 6.7.3.3. After turning power off, software must wait for at least 1 second before taking any action that relies on power having been removed from the hot-plug slot. For example, software is not permitted to turn off the power indicator (if present) or attempt to turn on the power controller before completing the 1 second wait period.

6.7.2. Registers Grouped by Hot-Plug Element Association

The registers described in this section are grouped by hot-plug element to convey all registers associated with implementing each element. Registers associated with each Downstream Port implementing a hot-plug capable slot are located in the Device Capabilities, Slot Capabilities, Slot Control, and Slot Status registers in the PCI Express Capability structure (see Section 7.8). Registers reporting the presence of hot-plug elements associated with the device Function on an adapter are located in the Device Capabilities register (also in the PCI Express Capability structure).

6.7.2.1. Attention Button Registers

Attention Button Present (Slot Capabilities and Device Capabilities) – This bit indicates if an Attention Button is electrically controlled by the chassis (Slot Capabilities) or by the adapter (Device Capabilities).

Attention Button Pressed (Slot Status) – This bit is set when an Attention Button electrically controlled by the chassis is pressed.

Attention Button Pressed Enable (Slot Control) – When Set, this bit enables software notification on an Attention Button Pressed event (see Section 6.7.3.4).

6.7.2.2. Attention Indicator Registers

Attention Indicator Present (Slot Capabilities and Device Capabilities) – This bit indicates if an Attention Indicator is electrically controlled by the chassis (Slot Capabilities) or by the adapter (Device Capabilities).

Attention Indicator Control (Slot Control) – When written, sets an Attention Indicator electrically controlled by the chassis to the written state.

6.7.2.3. Power Indicator Registers

Power Indicator Present (Slot Capabilities and Device Capabilities) – This bit indicates if a Power Indicator is electrically controlled by the chassis (Slot Capabilities) or by the adapter (Device Capabilities).

Power Indicator Control (Slot Control) – When written, sets a Power Indicator electrically controlled by the chassis to the written state.

6.7.2.4. Power Controller Registers

Power Controller Present (Slot Capabilities) – This bit indicates if a Power Controller is implemented.

Power Controller Control (Slot Control) – Turns the Power Controller on or off according to the value written.

Power Fault Detected (Slot Status) – This bit is set when a power fault is detected at the slot or the adapter.

Power Fault Detected Enable (Slot Control) – When Set, this bit enables software notification on a power fault event (see Section 6.7.3.4).

6.7.2.5. Presence Detect Registers

Presence Detect State (Slot Status) – This bit indicates the presence of an adapter in the slot.

Presence Detect Changed (Slot Status) – This bit is set when a presence detect state change is detected.

Presence Detect Changed Enable (Slot Control) – When Set, this bit enables software notification on a presence detect changed event (see Section 6.7.3.4).

6.7.2.6. MRL Sensor Registers

MRL Sensor Present (Slot Capabilities) – This bit indicates if an MRL Sensor is implemented.

MRL Sensor Changed (Slot Status) – This bit is set when the value of the MRL Sensor state changes.

MRL Sensor Changed Enable (Slot Control) – When Set, this bit enables software notification on a MRL Sensor changed event (see Section 6.7.3.4).

MRL Sensor State (Slot Status) – This register reports the status of the MRL Sensor if one is implemented.

6.7.2.7. Electromechanical Interlock Registers

Electromechanical Interlock Present (Slot Capabilities) – This bit indicates if an Electromechanical Interlock is implemented.

Electromechanical Interlock Status (Slot Status) – This bit reflects the current state of the Electromechanical Interlock.

Electromechanical Interlock Control (Slot Control) – This bit when set to 1b toggles the state of the Electromechanical Interlock.

6.7.2.8. Command Completed Registers

No Command Completed Support (Slot Capabilities) – This bit when set to 1b indicates that this slot does not generate software notification when an issued command is completed by the Hot-Plug Controller.

Command Completed (Slot Status) – This bit is set when the Hot-Plug Controller completes an issued command and is ready to accept the next command.

Command Completed Interrupt Enable (Slot Control) – When Set, this bit enables software notification (see Section 6.7.3.4) when a command is completed by the hot-plug control logic.

6.7.2.9. Port Capabilities and Slot Information Registers

Slot Implemented (PCI Express Capabilities) – When Set, this bit indicates that the Link associated with this Downstream Port is connected to a slot.

Physical Slot Number (Slot Capabilities) – This hardware initialized field indicates the physical slot number attached to the Port.

Hot-Plug Capable (Slot Capabilities) – When Set, this bit indicates this slot is capable of supporting hot-plug.

Hot-Plug Surprise (Slot Capabilities) – When Set, this bit indicates that adapter removal from the system without any prior notification is permitted for the associated form factor.

6.7.2.10. Hot-Plug Interrupt Control Register

Hot-Plug Interrupt Enable (Slot Control) – When Set, this bit enables generation of the hot-plug interrupt on enabled hot-plug events.

6.7.3. PCI Express Hot-Plug Events

A Downstream Port with hot-plug capabilities supports the following hot-plug events:

- Slot Events:
 - Attention Button Pressed
 - Power Fault Detected
 - MRL Sensor Changed
 - Presence Detect Changed
- Command Completed Events
- Data Link Layer State Changed Events

Each of these events has a status field, which indicates that an event has occurred but has not yet been processed by software, and an enable field, which indicates whether the event is enabled for software notification. Some events also have a capability field, which indicates whether the event type is supported on the Port. The grouping of these fields by event type is listed in Section 6.7.2, and each individual field is described in Section 7.8.

6.7.3.1. Slot Events

A Downstream Port with hot-plug capabilities monitors the slot it controls for the slot events listed above. When one of these slot events is detected, the Port indicates that the event has occurred by setting the status field associated with the event. At that point, the event is pending until software clears the status field.

Once a slot event is pending on a particular slot, all subsequent events of that type are ignored on that slot until the event is cleared. The Port must continue to monitor the slot for all other slot event types and report them as they occur.

If enabled through the associated enable field, slot events must generate a software notification. If the event is not supported on the Port as indicated by the associated capability field, software must not enable software notification for the event. The mechanism by which this notification is reported to software is described in Section 6.7.3.4.

6.7.3.2. Command Completed Events

Since changing the state of some hot-plug elements may not happen instantaneously, PCI Express supports hot-plug commands and command completed events. All hot-plug capable Ports are required to support hot-plug commands and, if the capability is reported, command completed events.

Software issues a command to a hot-plug capable Downstream Port by issuing a write transaction that targets any portion of the Port's Slot Control register. A single write to the Slot Control register is considered to be a single command, even if the write affects more than one field in the Slot Control register. In response to this transaction, the Port must carry out the requested actions and then set the associated status field for the command completed event. The Port must process the command normally even if the status field is already set when the command is issued. If a single command results in more than one action being initiated, the order in which the actions are executed is unspecified. All actions associated with a single command execution must not take longer than 1 second.

If command completed events are not supported as indicated by a value of 1b in the No Command Completed Support field of the Slot Capabilities register, a hot-plug capable Port must process a write transaction that targets any portion of the Port's Slot Control register without any dependency on previous Slot Control writes. Software is permitted to issue multiple Slot Control writes in sequence without any delay between the writes.

If command completed events are supported, then software must wait for a command to complete before issuing the next command. However, if the status field is not set after the 1 second limit on command execution, software is permitted to repeat the command or to issue the next command. If software issues a write before the Port has completed processing of the previous command and before the 1 second time limit has expired, the Port is permitted to either accept or discard the write. Such a write is considered a programming error, and could result in a discrepancy between the Slot Control register and the hot plug element state. To recover from such a programming error and return the controller to a consistent state, software must issue a write to the Slot Control register which conforms to the command completion rules.

If enabled through the associated enable field, the completion of a commands must generate a software notification. The exception to this rule is a command that occurs as a result of a write to the Slot Control register that disables software notification of command completed events. Such a command must be processed as described above, but must not generate a software notification.

6.7.3.3. Data Link Layer State Changed Events

The Data Link Layer State Changed event provides an indication that the state of the Data Link Layer Link Active bit in the Link Status register has changed. Support for Data Link Layer State Changed events and software notification of these events are required for hot-plug capable Downstream Ports. If this event is supported, the Port sets the status field associated with the event when the value in the Data Link Layer Link Active bit changes.

This event allows software to indirectly determine when power has been applied to a newly hot-plugged adapter. Software must wait for 100 ms after the Data Link Layer Link Active bit reads 1b before initiating a configuration access to the hot added device (see Section 6.6). Software must allow 1 second after the Data Link Layer Link Active bit reads 1b before it is permitted to determine that a hot plugged device which fails to return a Successful Completion for a Valid Configuration Request is a broken device (see Section 6.6).

The Data Link Layer State Changed event must occur within 1 second of the event that initiates the hot-insertion. If a power controller is supported, the time out interval is measured from when software initiated a write to the Slot Control register to turn on the power. If a power controller is not supported, the time out interval is measured from presence detect slot event. Software is allowed to time out on a hot add operation if the Data Link Layer State Changed event does not occur within 1 second. The action taken by software after such a timeout is implementation specific.

6.7.3.4. Software Notification of Hot-Plug Events

A hot-plug capable Downstream Port must support generation of an interrupt on a hot-plug event. As described in Sections 6.7.3.1 and 6.7.3.2, each hot-plug event has both an enable bit for interrupt generation and a status bit that indicates when an event has occurred but has not yet been processed by software. There is also a Hot-Plug Interrupt Enable bit in the Slot Control register that serves as a master enable/disable bit for all hot-plug events.

If the Port is enabled for level-triggered interrupt signaling using the INTx messages, the virtual INTx wire must be asserted whenever and as long as the following conditions are satisfied:

- The Interrupt Disable bit in the Command register is set to 0b.
- The Hot-Plug Interrupt Enable bit in the Slot Control register is set to 1b.
- At least one hot-plug event status bit in the Slot Status register and its associated enable bit in the Slot Control register are both set to 1b.

Note that all other interrupt sources within the same Function will assert the same virtual INTx wire when requesting service.

If the Port is enabled for edge-triggered interrupt signaling using MSI or MSI-X, an interrupt message must be sent every time the logical AND of the following conditions transitions from FALSE to TRUE:

- ❑ The associated vector is unmasked (not applicable if MSI does not support PVM).
- ❑ The Hot-Plug Interrupt Enable bit in the Slot Control register is set to 1b.
- ❑ At least one hot-plug event status bit in the Slot Status register and its associated enable bit in the Slot Control register are both set to 1b.

Note that PME and Hot-Plug Event interrupts (when both are implemented) always share the same MSI or MSI-X vector, as indicated by the Interrupt Message Number field in the PCI Express Capabilities register.

The Port may optionally send an MSI when there are hot-plug events that occur while interrupt generation is disabled, and interrupt generation is subsequently enabled.

If wake generation is required by the associated form factor specification, a hot-plug capable Downstream Port must support generation of a wakeup event (using the PME mechanism) on hot-plug events that occur when the system is in a sleep state or the Port is in device state D1, D2, or $D3_{Hot}$.

Software enables a hot-plug event to generate a wakeup event by enabling software notification of the event as described in Section 6.7.3.1. Note that in order for software to disable interrupt generation while keeping wakeup generation enabled, the Hot-Plug Interrupt Enable bit must be cleared. For form factors that support wake generation, a wakeup event must be generated if all three of the following conditions occur:

- ❑ The status register for an enabled event transitions from not set to set
- ❑ The Port is in device state D1, D2, or $D3_{Hot}$, and
- ❑ The PME_En bit in the Port's Power Management Control/Status register is set

Note that the Hot-Plug Controller generates the wakeup on behalf of the hot-plugged device, and it is not necessary for that device to have auxiliary (or main) power.

6.7.4. Firmware Support for Hot-Plug

Some systems that include hot-plug capable Root Ports and Switches that are released before ACPI-compliant operating systems with native hot-plug support are available, can use ACPI firmware for propagating hot-plug events. Firmware control of the hot-plug registers must be disabled if an operating system with native support is used. Platforms that provide ACPI firmware to propagate hot-plug events must also provide a mechanism to transfer control to the operating system. The details of this method are described in the *PCI Firmware Specification*.

6.7.5. Async Removal

Comment [VS212]: ECN: Downstream Port containment (DPC)

Async removal refers to the removal of an adapter or disabling of a Downstream Port Link due to error containment without prior warning to the operating system. This is contrast to standard PCI hot-plug where removal operations are performed in a lock-step manner with the OS through a well defined sequence of user actions and system management facilities. For example, the user presses the Attention Button to request permission from the OS to remove the adapter, but the user doesn't actually remove the adapter from the slot until the OS has quiesced activity to the adapter and granted permission for removal.

Since async removal proceeds before the rest of the PCI Express hierarchy or OS necessarily becomes aware of the event, special consideration is required beyond that needed for standard PCI hot-plug. This section outlines PCI Express events that may occur as a side effect of async removal and mechanisms for handling async removal.

Since async removal may be unexpected to both the Physical and Data Link Layers of the Downstream Port associated with the slot, Correctable Errors may be reported as a side effect of the event (i.e., Receiver Error, Bad TLP, and Bad DLLP). If these errors are reported, software should handle them as an expected part of this event.

Requesters may experience Completion Timeouts associated with Requests that were accepted, but will never be completed by removed Completers. Any resulting Completion Timeout errors in this context should be handled as an expected part of this event.

Async removal may result in a transition from DL_Active to DL_Down in the Downstream port. This transition may result in a Surprise Down error. In addition, Requesters in the PCI Express hierarchy domain may not become immediately aware of this transition and continue to issue Requests to removed Completers that must be handled by the Downstream Port associated with the slot.

The Surprise Down error resulting from async removal may trigger Downstream Port Containment (See Section 6.2.10). Downstream Port Containment following an async removal may be utilized to hold the Link of a Downstream Port in the Disabled LTSSM state while host software recovers from the side effects of an async removal.

6.8. Power Budgeting Capability

With the addition of a hot-plug capability for adapters, the need arises for the system to be capable of properly allocating power to any new devices added to the system. This capability is a separate and distinct function from power management and a basic level of support is required to ensure proper operation of the system. The power budgeting concept puts in place the building blocks that allow devices to interact with systems to achieve these goals. There are many ways in which the system can implement the actual power budgeting capabilities, and as such, they are beyond the scope of this specification.

Implementation of the Power Budgeting Capability is optional for devices that are implemented either in a form factor which does not require hot-plug support, or that are integrated on the system board. Form factor specifications may require support for power budgeting. The devices and/or adapters are required to remain under the configuration power limit specified in the corresponding electromechanical specification until they have been configured and enabled by the system. The system should guarantee that power has been properly budgeted prior to enabling an adapter.

6.8.1. System Power Budgeting Process Recommendations

It is recommended that system firmware provide the power budget management agent the following information:

- Total system power budget (power supply information).
- Total power allocated by system firmware (system board devices).
- Total number of slots and the types of slots.

System firmware is responsible for allocating power for all devices on the system board that do not have power budgeting capabilities. The firmware may or may not include devices that are connected to the standard power rails. When the firmware allocates the power for a device that implements the Power Budgeting Capability it must set the System Allocated bit to 1b in the Power Budget Capability register to indicate that it has been properly allocated. The power budget manager is responsible for allocating all PCI Express devices including system board devices that have the Power Budgeting Capability and have the System Allocated bit Clear. The power budget manager is responsible for determining if hot-plugged devices can be budgeted and enabled in the system.

There are alternate methods which may provide the same functionality, and it is not required that the power budgeting process be implemented in this manner.

6.9. Slot Power Limit Control

PCI Express provides a mechanism for software controlled limiting of the maximum power per slot that a PCI Express adapter (associated with that slot) can consume. The key elements of this mechanism are:

- Slot Power Limit Value and Scale fields of the Slot Capabilities register implemented in the Downstream Ports of a Root Complex or a Switch
- Captured Slot Power Limit Value and Scale fields of the Device Capabilities register implemented in Endpoint, Switch, or PCI Express-PCI Bridge Functions present in an Upstream Port
- Set_Slot_Power_Limit Message that conveys the content of the Slot Power Limit Value and Scale fields of the Slot Capabilities register of the Downstream Port (of a Root Complex or a Switch) to the corresponding Captured Slot Power Limit Value and Scale fields of the Device Capabilities register in the Upstream Port of the component connected to the same Link

Power limits on the platform are typically controlled by the software (for example, platform firmware) that comprehends the specifics of the platform such as:

- Partitioning of the platform, including slots for I/O expansion using adapters
- Power delivery capabilities
- Thermal capabilities

This software is responsible for correctly programming the Slot Power Limit Value and Scale fields of the Slot Capabilities registers of the Downstream Ports connected to slots. After the value has been written into the register within the Downstream Port, it is conveyed to the adapter using the Set_Slot_Power_Limit Message (see Section 2.2.8.5). The recipient of the Message must use the value in the Message data payload to limit usage of the power for the entire adapter, unless the adapter will never exceed the lowest value specified in the corresponding form factor specification. It is required that device driver software associated with the adapter be able (by reading the values of the Captured Slot Power Limit Value and Scale fields of the Device Capabilities register) to configure hardware of the adapter to guarantee that the adapter will not exceed the imposed limit. In the case where the platform imposes a limit that is below the minimum needed for adequate operation, the device driver will be able to communicate this discrepancy to higher level configuration software. Configuration software is required to set the Slot Power Limit to one of the maximum values specified for the corresponding form factor based on the capability of the platform.

The following rules cover the Slot Power Limit control mechanism:

For Adapters:

- Until and unless a Set_Slot_Power_Limit Message is received indicating a Slot Power Limit value greater than the lowest value specified in the form factor specification for the adapter's form factor, the adapter must not consume more than the lowest value specified.
- An adapter must never consume more power than what was specified in the most recently received Set_Slot_Power_Limit Message or the minimum value specified in the corresponding form factor specification, whichever is higher.

- Components with Endpoint, Switch, or PCI Express-PCI Bridge Functions that are targeted for integration on an adapter where total consumed power is below the lowest limit defined for the targeted form factor are permitted to ignore Set_Slot_Power_Limit Messages, and to return a value of 0 in the Captured Slot Power Limit Value and Scale fields of the Device Capabilities register
 - Such components still must be able to receive the Set_Slot_Power_Limit Message without error but simply discard the Message value

For Root Complex and Switches which source slots:

- Configuration software must not program a Set_Slot_Power_Limit value that indicates a limit that is lower than the lowest value specified in the form factor specification for the slot's form factor.

IMPLEMENTATION NOTE

Example Adapter Behavior Based on the Slot Power Limit Control Capability

The following power limit scenarios are examples of how an adapter must behave based on the Slot Power Limit control capability. The form factor limits are representations, and should not be taken as actual requirements.

Note: Form factor #1 has a maximum power requirement of 40 W and 25 W; form factor #2 has a maximum power requirement of 15 W.

Scenario 1: An Adapter Consuming 12 W

- If the adapter is plugged into a form factor #1 40 W slot, the Slot Power Limit control mechanism is followed, and the adapter operates normally.
- If the adapter is plugged into a form factor #1 25 W slot, the Slot Power Limit control mechanism is followed, and the adapter operates normally.
- If the adapter is plugged into a form factor #2 15 W slot, the Slot Power Limit control mechanism is followed, and the adapter operates normally.

In all cases, since the adapter operates normally within all the form factors, it can ignore any of the slot power limit Messages.

Scenario 2: An Adapter Consuming 18 W

- If the adapter is plugged into a form factor #1 40 W slot, the Slot Power Limit control mechanism is followed, and the adapter operates normally.
- If the adapter is plugged into a form factor #1 25 W slot, the Slot Power Limit control mechanism is followed, and the adapter operates normally.
- If the adapter is plugged into a form factor #2 15 W slot, the Slot Power Limit control mechanism is followed, and the adapter must scale down to 15 W or disable operation. An adapter that does not scale within any of the power limits for a given form factor will always be disabled in that form factor and should not be used.

In this case, if the adapter is only to be used in form factor #1, it can ignore any of the slot power limit Messages. To be useful in form factor #2, the adapter should be capable of scaling to the power limit of form factor #2.

Scenario 3: An Adapter Consuming 30 W

- If the adapter is plugged into a form factor #1 40 W slot, the Slot Power Limit control mechanism is followed, and the device operates normally.
- If the adapter is plugged into a form factor #1 25 W slot, the Slot Power Limit control mechanism is followed, and the device must scale down to 25 W or disable operation.
- If the adapter is plugged into a form factor #2 15 W slot, the Slot Power Limit control mechanism is followed, and the adapter must scale down to 15 W or disable operation. An adapter that does not scale within any of the power limits for a given form factor will always be disabled in that form factor and should not be used.

In this case, since the adapter consumes power above the lowest power limit for a slot, the adapter must be capable of scaling or disabling to prevent system failures. Operation of adapters at power levels that exceed the capabilities of the slots in which they are plugged must be avoided.

IMPLEMENTATION NOTE

Slot Power Limit Control Registers

Typically Slot Power Limit register fields within Downstream Ports of a Root Complex or a Switch will be programmed by platform-specific software. Some implementations may use a hardware method for initializing the values of these registers and, therefore, do not require software support.

Components with Endpoint, Switch, or PCI Express-PCI Bridge Functions that are targeted for integration on the adapter where total consumed power is below the lowest limit defined for that form factor are allowed to ignore Set_Slot_Power_Limit Messages. Note that components that take this implementation approach may not be compatible with potential future defined form factors. Such form factors may impose lower power limits that are below the minimum required by a new adapter based on the existing component.

IMPLEMENTATION NOTE

Auto Slot Power Limit Disable

In some environments host software may wish to directly manage the transmission of a Set_Slot_Power_Limit message by performing a Configuration Write to the Slot Capabilities register rather than have the transmission automatically occur when the Link transitions from a non-DL_Up to a DL_Up status. This allows host software to limit power supply surge current by staggering the transition of Endpoints to a higher power state following a Link Down or when multiple Endpoints are simultaneously hot-added due to cable or card insertion.

Formatted: Font color: Black

Comment [VS213]: ECN: Downstream Port containment (DPC)

6.10. Root Complex Topology Discovery

A Root Complex may present one of the following topologies to configuration software:

- A single opaque Root Complex such that software has no visibility with respect to internal operation of the Root Complex. All Root Ports are independent of each other from a software perspective; no mechanism exists to manage any arbitration among the various Root Ports for any differentiated services.
- A single Root Complex Component such that software has visibility and control with respect to internal operation of the Root Complex Component. As shown in Figure 6-11, software views the Root Ports as Ingress Ports for the component. The Root Complex internal Port for traffic aggregation to a system Egress Port or an internal sink unit (such as memory) is represented by an RCRB structure. Controls for differentiated services are provided through a Virtual Channel Capability structure located in the RCRB.

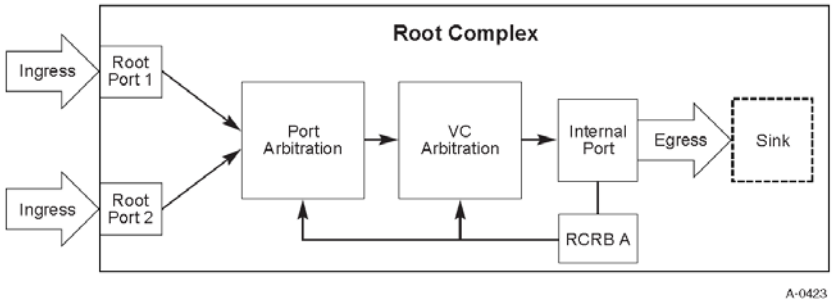


Figure 6-11: Root Complex Represented as a Single Component

- Multiple Root Complex Components such that software not only has visibility and control with respect to internal operation of a given Root Complex Component but also has the ability to discover and control arbitration between different Root Complex Components. As shown in Figure 6-12, software views the Root Ports as Ingress Ports for a given component. An RCRB structure controls egress from the component to other Root Complex Components (RCRB C) or to an internal sink unit such as memory (RCRB A). In addition, an RCRB structure (RCRB B) may also be present in a given component to control traffic from other Root Complex Components. Controls for differentiated services are provided through Virtual Channel Capability structures located appropriately in the RCRBs respectively.

More complex topologies are possible as well.

A Root Complex topology can be represented as a collection of logical Root Complex Components such that each logical component has:

- One or more Ingress Ports.
- An Egress Port.
- Optional associated Virtual Channel capabilities located either in the Configuration Space (for Root Ports) or in an RCRB (for internal Ingress/Egress Ports) if the Root Complex supports Virtual Channels.
- Optional devices/Functions integrated in the Root Complex.

In order for software to correctly program arbitration and other control parameters for PCI Express differentiated services, software must be able to discover a Root Complex's internal topology. Root Complex topology discovery is accomplished by means of the Root Complex Link Declaration Capability as described in Section 7.13.

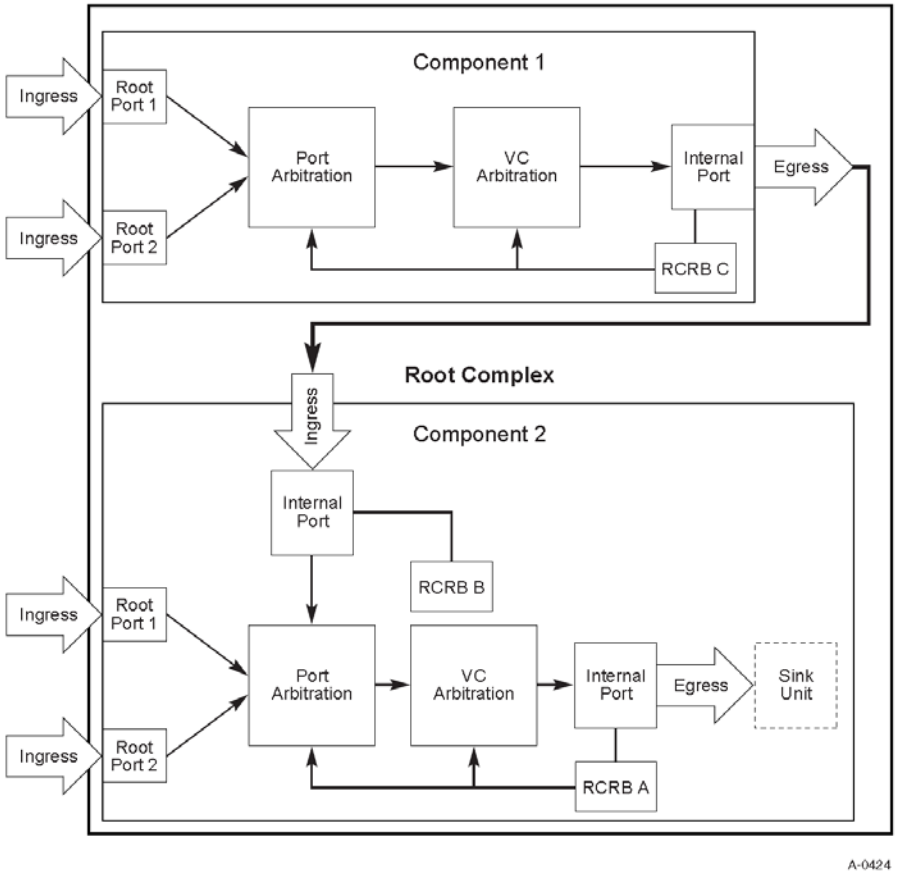


Figure 6-12: Root Complex Represented as Multiple Components

6.11. Link Speed Management

This section describes how Link speed management is coordinated between the LTSSM (Section 4.2.6) and the software Link observation and control mechanisms (Sections 7.8.6, 7.8.7, 7.8.8, 7.8.18, 7.8.19, and 7.8.20).

The Target Link Speed field in the Link Control 2 register sets the upper bound for the Link speed. Except as described below, the Upstream component must attempt to maintain the Link at the Target Link Speed, or at the highest speed supported by both components on the Link (as reported by the values in the training sets – see Section 4.2.4.1), whichever is lower.

If the Hardware Autonomous Speed Disable bit in the Link Control 2 register is clear, the component is permitted to autonomously adjust the Link speed using implementation specific criteria.

If the reliability of the Link is unacceptably low, then either component is permitted to lower the Link speed by removing the unreliable Link speed from the list of supported speeds advertised in the training sets the component transmits. The criteria for determination of acceptable Link reliability are implementation specific, and are not dependent on the setting of the Hardware Autonomous Speed Disable bit.

During any given speed negotiation it is possible that one or both components will advertise a subset of all speeds supported, as a means to cap the post-negotiation Link speed. It is permitted for a component to change its set of advertised supported speeds without requesting a Link speed change by driving the Link through Recovery without setting the speed change bit.

When a component's attempt to negotiate to a particular Link speed fails, that component is not permitted to attempt negotiation to that Link speed, or to any higher Link speed, until 200 ms has passed from the return to L0 following the failed attempt, or until the other component on the Link advertises support for the higher Link speed through its transmitted training sets (with or without a request to change the Link speed), whichever comes first.

Software is permitted to restrict the maximum speed of Link operation and set the preferred Link speed by setting the value in the Target Link Speed field in the Upstream component. After modifying the value in the Target Link Speed field, software must trigger Link retraining by writing 1b to the Retrain Link bit. Software is notified of any Link speed changes (as well as any Link width changes) through the Link Bandwidth Notification Mechanism.

Software is permitted to cause a Link to transition to the Polling.Compliance LTSSM state by writing to the Target Link Speed field and setting the Enter Compliance bit in the Link Control 2 register in both components, and then initiating a Hot Reset on the Link (through the Upstream component). Software is required to write the same value into the Target Link Speed field in both the Upstream and Downstream components. Note that this will take the Link to a DL_Down state and therefore cannot be done transparently to other software that is using the Link. The Downstream Port will return to Polling.Active when the Enter Compliance bit is cleared.

6.12. Access Control Services (ACS)

ACS defines a set of control points within a PCI Express topology to determine whether a TLP should be routed normally, blocked, or redirected. ACS is applicable to RCs, Switches, and multi-Function devices.[90]

ACS provides the following types of access control:

- ACS Source Validation (V)
- ACS Translation Blocking (B)
- ACS P2P Request Redirect (R)
- ACS P2P Completion Redirect (C)
- ACS Upstream Forwarding (U)
- ACS P2P Egress Control (E)

[90] Applicable Functions within multi-Function devices specifically include PCI Express Endpoints, Switch Upstream Ports, Legacy PCI Express Endpoints, and Root Complex Integrated Endpoints.

- ACS Direct Translated P2P (T)

The specific requirements for each of these are discussed in the following section. The letter in parenthesis following each type is the abbreviation for the associated capability and control bits defined in Section 7.16.

ACS hardware functionality is disabled by default, and is enabled only by ACS-aware software. With the exception of ACS Source Validation, ACS access controls are not applicable to Multicast TLPs (see Section 6.14), and have no effect on them.

6.12.1. ACS Component Capability Requirements

ACS functionality is reported and managed via ACS Extended Capability structures. PCI Express components are permitted to implement ACS Extended Capability structures in some, none, or all of their applicable Functions. The extent of what is implemented is communicated through capability bits in each ACS Extended Capability structure. A given Function with an ACS Extended Capability structure may be required or forbidden to implement certain capabilities, depending upon the specific type of the Function and whether it is part of a multi-Function device.

ACS is never applicable to a PCI Express to PCI Bridge Function or a Root Complex Event Collector Function, and such Functions must never implement an ACS Extended Capability structure.

6.12.1.1. ACS Downstream Ports

This section applies to Root Ports and Downstream Switch Ports that implement an ACS Extended Capability structure. This section applies to Downstream Port Functions both for single-Function devices and multi-Function devices.

- ❑ ACS Source Validation: must be implemented.

 When enabled, the Downstream Port tests the Bus Number from the Requester ID of each Upstream Request received by the Port to determine if it is within the Bus Number "aperture" of the Port – the inclusive range specified by the Secondary Bus Number register and the Subordinate Bus Number register.

 If the Bus Number from the Requester ID of the Request is not within this aperture, this is a reported error (ACS Violation) associated with the Receiving Port (see Section 6.12.4.)

 Completions are never affected by ACS Source Validation.

- ❑ ACS Translation Blocking: must be implemented.

 When enabled, the Downstream Port checks the Address Translation (AT) field of each Upstream Memory Request received by the Port. If the AT field is not the default value, this is a reported error (ACS Violation) associated with the Receiving Port (see Section 6.12.4). This error must take precedence over ACS Upstream Forwarding and any applicable ACS P2P control mechanisms.

 Completions are never affected by ACS Translation Blocking.

- ❑ ACS P2P Request Redirect: must be implemented by Root Ports that support peer-to-peer traffic with other Root Ports;[91] must be implemented by Switch Downstream Ports.

 ACS P2P Request Redirect is subject to interaction with the ACS P2P Egress Control and ACS Direct Translated P2P mechanisms (if implemented). Refer to Section 6.12.3 for more information.

 When ACS P2P Request Redirect is enabled in a Switch Downstream Port, peer-to-peer Requests must be redirected Upstream towards the RC.

 When ACS P2P Request Redirect is enabled in a Root Port, peer-to-peer Requests must be sent to Redirected Request Validation logic within the RC that determines whether the Request is "reflected" back Downstream towards its original target, or blocked as an ACS Violation error. The algorithms and specific controls for making this determination are not architected by this specification.

 Downstream Ports never redirect Requests that are traveling Downstream.

 Completions are never affected by ACS P2P Request Redirect.

- ❑ ACS P2P Completion Redirect: must be implemented by Root Ports that implement ACS P2P Request Redirect; must be implemented by Switch Downstream Ports.

[91] Root Port indication of ACS P2P Request Redirect or ACS P2P Completion Redirect support does not imply any particular level of peer-to-peer support by the Root Complex, or that peer-to-peer traffic is supported at all

The intent of ACS P2P Completion Redirect is to avoid ordering rule violations between Completions and Requests when Requests are redirected. Refer to Section 6.12.5 for more information.

ACS P2P Completion Redirect does not interact with ACS controls that govern Requests.

When ACS P2P Completion Redirect is enabled in a Switch Downstream Port, peer-to-peer Completions[92] that do not have the Relaxed Ordering Attribute bit set (1b) must be redirected Upstream towards the RC. Otherwise, peer-to-peer Completions must be routed normally.

When ACS P2P Completion Redirect is enabled in a Root Port, peer-to-peer Completions that do not have the Relaxed Ordering bit set must be handled such that they do not pass Requests that are sent to Redirected Request Validation logic within the RC. Such Completions must eventually be sent Downstream towards their original peer-to-peer targets, without incurring additional ACS access control checks.

Downstream Ports never redirect Completions that are traveling Downstream.

Requests are never affected by ACS P2P Completion Redirect.

- ❑ ACS Upstream Forwarding: must be implemented by Root Ports if the RC supports Redirected Request Validation; must be implemented by Switch Downstream Ports.

 When ACS Upstream Forwarding is enabled in a Switch Downstream Port, and its Ingress Port receives an Upstream Request or Completion TLP targeting the Port's own Egress Port, the Port must instead forward the TLP Upstream towards the RC.

 When ACS Upstream Forwarding is enabled in a Root Port, and its Ingress Port receives an Upstream Request or Completion TLP that targets the Port's own Egress Port, the Port must handle the TLP as follows. For a Request, the Root Port must handle it the same as a Request that the Port "redirects" with the ACS P2P Request Redirect mechanism. For a Completion, the Root Port must handle it the same as a Completion that the Port "redirects" with the ACS P2P Completion Redirect mechanism.

 When ACS Upstream Forwarding is not enabled on a Downstream Port, and its Ingress Port receives an Upstream Request or Completion TLP that targets the Port's own Egress Port, the handling of the TLP is undefined.

- ❑ ACS P2P Egress Control: implementation is optional.

 ACS P2P Egress Control is subject to interaction with the ACS P2P Request Redirect and ACS Direct Translated P2P mechanisms (if implemented). Refer to Section 6.12.3 for more information.

 A Switch that supports ACS P2P Egress Control can be selectively configured to block peer-to-peer Requests between its Downstream Ports. Software can configure the Switch to allow none or only a subset of its Downstream Ports to send peer-to-peer Requests to other Downstream Ports. This is configured on a per Downstream Port basis.

 An RC that supports ACS P2P Egress Control can be selectively configured to block peer-to-peer Requests between its Root Ports. Software can configure the RC to allow none or only a

[92] This includes Read Completions, AtomicOp Completions, and other Completions with or without Data.

subset of the ~~Root Port hierarchies~~ Hierarchy Domains to send peer-to-peer Requests to other ~~Root Port hierarchies~~ Hierarchy Domains. This is configured on a per Root Port basis.

Comment [VS214]: Errata A20

With ACS P2P Egress Control in Downstream Ports, controls in the Ingress Port ("sending" Port) determine if the peer-to-peer Request is blocked, and if so, the Ingress Port handles the ACS Violation error per Section 6.12.4.

Completions are never affected by ACS P2P Egress Control.

- ACS Direct Translated P2P: must be implemented by Root Ports that support Address Translation Services (ATS) and also support peer-to-peer traffic with other Root Ports;[93] must be implemented by Switch Downstream Ports.

 When ACS Direct Translated P2P is enabled in a Downstream Port, peer-to-peer Memory Requests whose Address Type (AT) field indicates a Translated address must be routed normally ("directly") to the peer Egress Port, regardless of ACS P2P Request Redirect and ACS P2P Egress Control settings. All other peer-to-peer Requests must still be subject to ACS P2P Request Redirect and ACS P2P Egress Control settings.

 Completions are never affected by ACS Direct Translated P2P.

6.12.1.2. ACS Functions in Multi-Function Devices

This section applies to multi-Function device ACS Functions, with the exception of Downstream Port Functions, which are covered in the preceding section.

- ACS Source Validation: must not be implemented.
- ACS Translation Blocking: must not be implemented.
- ACS P2P Request Redirect: must be implemented by Functions that support peer-to-peer traffic with other Functions.

 ACS P2P Request Redirect is subject to interaction with the ACS P2P Egress Control and ACS Direct Translated P2P mechanisms (if implemented). Refer to Section 6.12.3 for more information.

 When ACS P2P Request Redirect is enabled in a multi-Function device, peer-to-peer Requests (between Functions of the device) must be redirected Upstream towards the RC.

 Completions are never affected by ACS P2P Request Redirect.

- ACS P2P Completion Redirect: must be implemented by Functions that implement ACS P2P Request Redirect.

 The intent of ACS P2P Completion Redirect is to avoid ordering rule violations between Completions and Requests when Requests are redirected. Refer to Section 6.12.5 for more information.

 ACS P2P Completion Redirect does not interact with ACS controls that govern Requests.

[93] Root Port indication of ACS Direct Translated P2P support does not imply any particular level of peer-to-peer support by the Root Complex, or that peer-to-peer traffic is supported at all.

When ACS P2P Completion Redirect is enabled in a multi-Function device, peer-to-peer Read Completions that do not have the Relaxed Ordering bit set must be redirected Upstream towards the RC. Otherwise, peer-to-peer Completions must be routed normally.

Requests are never affected by ACS P2P Completion Redirect.

- ACS Upstream Forwarding: must not be implemented.
- ACS P2P Egress Control: implementation is optional; is based on Function Numbers or Function Group Numbers; controls peer-to-peer Requests between the different Functions within the multi-Function device.

 ACS P2P Egress Control is subject to interaction with the ACS P2P Request Redirect and ACS Direct Translated P2P mechanisms (if implemented). Refer to Section 6.12.3 for more information.

 Each Function within a multi-Function device that supports ACS P2P Egress Control can be selectively enabled to block peer-to-peer communication with other Functions or Function Groups[94] within the device. This is configured on a per Function basis.

 With ACS P2P Egress Control in multi-Function devices, controls in the "sending" Function determine if the Request is blocked, and if so, the "sending" Function handles the ACS Violation error per Section 6.12.4.

 When ACS Function Groups are enabled in an ARI Device, ACS P2P Egress Controls are enforced on a per Function Group basis instead of a per Function basis. See Section 6.13.

 Completions are never affected by ACS P2P Egress Control.

- ACS Direct Translated P2P: must be implemented if the multi-Function device Function supports Address Translation Services (ATS) and also peer-to-peer traffic with other Functions.

 When ACS Direct Translated P2P is enabled in a multi-Function device Function, peer-to-peer Memory Requests whose Address Type (AT) field indicates a Translated address must be routed normally ("directly") to the peer Function, regardless of ACS P2P Request Redirect and ACS P2P Egress Control settings. All other peer-to-peer Requests must still be subject to ACS P2P Request Redirect and ACS P2P Egress Control settings.

 Completions are never affected by ACS Direct Translated P2P.

6.12.1.3. Functions in Single-Function Devices

This section applies to single-Function device Functions, with the exception of Downstream Port Functions, which are covered in a preceding section. No ACS capabilities are applicable, and the Function must not implement an ACS Extended Capability structure.

6.12.2. Interoperability

The following rules govern interoperability between ACS and non-ACS components:

[94] ACS Function Groups capability is optional for ARI Devices that implement ACS P2P Egress Controls.

- When ACS P2P Request Redirect and ACS P2P Completion Redirect are not being used, ACS and non-ACS components may be intermixed within a topology and will interoperate fully. ACS can be enabled in a subset of the ACS components without impacting interoperability.
- When ACS P2P Request Redirect, ACS P2P Completion Redirect, or both are being used, certain components in the PCI Express hierarchy must support ACS Upstream Forwarding (of Upstream redirected Requests). Specifically:

The associated Root Port[95] must support ACS Upstream Forwarding. Otherwise, how the Root Port handles Upstream redirected Request or Completion TLPs is undefined. The RC must also implement Redirected Request Validation.

Between each ACS component where P2P TLP redirection is enabled and its associated Root Port, any intermediate Switches must support ACS Upstream Forwarding. Otherwise, how such Switches handle Upstream redirected TLPs is undefined.

6.12.3. ACS Peer-to-Peer Control Interactions

With each peer-to-peer Request, multiple ACS control mechanisms may interact to determine whether the Request is routed directly towards its peer-to-peer target, blocked immediately as an ACS Violation, or redirected Upstream towards the RC for access validation. Peer-to-peer Completion redirection is determined exclusively by the ACS P2P Completion Redirect mechanism.

If ACS Direct Translated P2P is enabled in a Port/Function, peer-to-peer Memory Requests whose Address Translation (AT) field indicates a Translated address must be routed normally ("directly") to the peer Port/Function, regardless of ACS P2P Request Redirect and ACS P2P Egress Control settings. Otherwise such Requests, and unconditionally all other peer-to-peer Requests, must be subject to ACS P2P Request Redirect and ACS P2P Egress Control settings. Specifically, the applicable Egress Control Vector bit, along with the ACS P2P Egress Control Enable bit (E) and the ACS P2P Request Redirect Enable bit (R), determine how the Request is handled. It must be noted that atomicity of accesses cannot be guaranteed if ACS peer-to-peer Request Redirect targets a legacy device location that can be the target of a locked access. Refer to Section 7.16 for descriptions of these control bits. Table 6-9 specifies the interactions.

Table 6-9: ACS P2P Request Redirect and ACS P2P Egress Control Interactions

Control Bit E (b)	Control Bit R (b)	Egress Control Vector Bit for the Associated Egress Switch Port, Root Port, Function, or Function Group	Required Handling for Peer-to-Peer Requests
0	0	X – Don't care	Route directly to peer-to-peer target
0	1	X – Don't Care	Redirect Upstream
1	0	1	Handle as an ACS Violation
1	0	0	Route directly to peer-to-peer target
1	1	1	Redirect Upstream
1	1	0	Route directly to peer-to-peer target

[95] Not applicable for ACS Redirect between Functions of a multi-Function Root Complex Integrated Endpoint.

6.12.4. ACS Violation Error Handling

ACS Violations may occur due to either hardware or software defects/failures. To assist in fault isolation and root cause analysis, it is recommended that AER be implemented in ACS components. AER prefix/header logging and the Prefix Log/Header Log registers may be used to determine the prefix/header of the offending Request. The ACS Violation Status, Mask, and Severity bits provide positive identification of the error and increased control over error logging and signaling.

When an ACS Violation is detected, the ACS component that operates as the Completer[96] must do the following:

- For Non-Posted Requests, the Completer must generate a Completion with a Completer Abort (CA) Completion Status.
- The Completer must log and signal the ACS Violation as indicated in Figure 6-2. Note the following:
 - Even though the Completer uses a CA Completion Status when it sends a Completion, the Completer must log an ACS Violation error instead of a Completer Abort error.
 - If the severity of the ACS Violation is non-fatal and the Completer sends a Completion with CA Completion Status, this case must be handled as an Advisory Non-Fatal Error as described in Section 6.2.3.2.4.1.
- The Completer[97] must set the Signaled Target Abort bit in either its Status register or Secondary Status register as appropriate.

6.12.5. ACS Redirection Impacts on Ordering Rules

When ACS P2P Request Redirect is enabled, some or all peer-to-peer Requests are redirected, which can cause ordering rule violations in some cases. This section explores those cases, plus a similar case that occurs with RCs that implement "Request Retargeting" as an alternative mechanism for enforcing peer-to-peer access control.

6.12.5.1. Completions Passing Posted Requests

When a peer-to-peer Posted Request is redirected, a subsequent peer-to-peer non-RO[98] Completion that is routed directly can effectively pass the redirected Posted Request, violating the ordering rule that non-RO Completions must not pass Posted Requests. Refer to Section 2.4.1 for more information.

[96] In all cases but one, the ACS component that detects the ACS Violation also operates as the Completer. The exception case is when Root Complex Redirected Request Validation logic disallows a redirected Request. If the redirected Request came through a Root Port, that Root Port must operate as the Completer. If the redirected Request came from a Root Complex Integrated Endpoint, the associated Root Complex Event Collector must operate as the Completer.

[97] Similarly, if the Request was Non-Posted, when the Requester receives the resulting Completion with CA Completion Status, the Requester must set the Received Target Abort bit in either its Status register or Secondary Status register as appropriate. Note that for the case of a multi-Function device incurring an ACS Violation error with a peer-to-peer Request between its Functions, the same Function might serve both as Requester and Completer.

[98] In this section, "non-RO" is an abbreviation characterizing TLPs whose Relaxed Ordering Attribute field is not set.

ACS P2P Completion Redirect can be used to avoid violating this ordering rule. When ACS P2P Completion Redirect is enabled, all peer-to-peer non-RO Completions will be redirected, thus taking the same path as redirected peer-to-peer Posted Requests. Enabling ACS P2P Completion Redirect when some or all peer-to-peer Requests are routed directly will not cause any ordering rule violations, since it is permitted for a given Completion to be passed by any TLP other than another Completion with the same Transaction ID.

As an alternative mechanism to ACS P2P Request Redirect for enforcing peer-to-peer access control, some RCs implement "Request Retargeting", where the RC supports special address ranges for "peer-to-peer" traffic, and the RC will retarget validated Upstream Requests to peer devices. Upon receiving an Upstream Request targeting a special address range, the RC validates the Request, translates the address to target the appropriate peer device, and sends the Request back Downstream. With retargeted Requests that are Non-posted, if the RC does not modify the Requester ID, the resulting Completions will travel "directly" peer-to-peer back to the original Requester, creating the possibility of non-RO Completions effectively passing retargeted Posted Requests, violating the same ordering rule as when ACS P2P Request Redirect is being used. ACS P2P Completion Redirect can be used to avoid violating this ordering rule here as well.

If ACS P2P Request Redirect and RC P2P Request Retargeting are not being used, there is no envisioned benefit to enabling ACS P2P Completion Redirect, and it is recommended not to do so because of potential performance impacts.



IMPLEMENTATION NOTE

Performance Impacts with ACS P2P Completion Redirect

While the use of ACS P2P Completion Redirect can avoid ordering violations with Completions passing Posted Requests, it also may impact performance. Specifically, all redirected Completions will have to travel up to the RC from the point of redirection and back, introducing extra latency and possibly increasing Link and RC congestion.

Since peer-to-peer Completions with the Relaxed Ordering bit set are never redirected (thus avoiding performance impacts), it is strongly recommended that Requesters be implemented to maximize the proper use of Relaxed Ordering, and that software enable Requesters to utilize Relaxed Ordering by setting the Enable Relaxed Ordering bit in the Device Control register.

If software enables ACS P2P Request Redirect, RC P2P Request Retargeting, or both, and software is certain that proper operation is not compromised by peer-to-peer non-RO Completions passing peer-to-peer[99] Posted Requests, it is recommended that software leave ACS P2P Completion Redirect disabled as a way to avoid its performance impacts.

6.12.5.2. Requests Passing Posted Requests

When some peer-to-peer Requests are redirected but other peer-to-peer Requests are routed directly, the possibility exists of violating the ordering rules where Non-posted Requests or non-RO Posted Requests must not pass Posted Requests. Refer to Section 2.4.1 for more information.

These ordering rule violation possibilities exist only when ACS P2P Request Redirect and ACS Direct Translated P2P are both enabled. Software should not enable both these mechanisms unless it is certain either that such ordering rule violations cannot occur, or that proper operation will not be compromised if such ordering rule violations do occur.

[99] These include true peer-to-peer Requests that are redirected by the ACS P2P Request Redirect mechanism, as well as "logically peer-to-peer" Requests routed to the Root Complex that the Root Complex then retargets to the peer device.



IMPLEMENTATION NOTE

Ensuring Proper Operation with ACS Direct Translated P2P

The intent of ACS Direct Translated P2P is to optimize performance in environments where Address Translation Services (ATS) are being used with peer-to-peer communication whose access control is enforced by the RC. Permitting peer-to-peer Requests with Translated addresses to be routed directly avoids possible performance impacts associated with redirection, which introduces extra latency and may increase Link and RC congestion.

For the usage model where peer-to-peer Requests with Translated addresses are permitted, but those with Untranslated addresses are to be blocked as ACS Violations, it is recommended that software enable ACS Direct Translated P2P and ACS P2P Request Redirect, and configure the Redirected Request Validation logic in the RC to block the redirected Requests with Untranslated addresses. This configuration has no ordering rule violations associated with Requests passing Posted Requests.

For the usage model where some Requesters use Translated addresses exclusively with peer-to-peer Requests and some Requesters use Untranslated addresses exclusively with peer-to-peer Requests, and the two classes of Requesters do not communicate peer-to-peer with each other, proper operation is unlikely to be compromised by redirected peer-to-peer Requests (with Untranslated addresses) being passed by direct peer-to-peer Requests (with Translated addresses). It is recommended that software not enable ACS Direct Translated P2P unless software is certain that proper operation is not compromised by the resulting ordering rule violations.

For the usage model where a single Requester uses both Translated and Untranslated addresses with peer-to-peer Requests, again it is recommended that software not enable ACS Direct Translated P2P unless software is certain that proper operation is not compromised by the resulting ordering rule violations. This requires a detailed analysis of the peer-to-peer communications models being used, and is beyond the scope of this specification.

6.13. Alternative Routing-ID Interpretation (ARI)

Routing IDs, Requester IDs, and Completer IDs are 16-bit identifiers traditionally composed of three fields: an 8-bit Bus Number, a 5-bit Device Number, and a 3-bit Function Number. With ARI, the 16-bit field is interpreted as two fields instead of three: an 8-bit Bus Number and an 8-bit Function Number – the Device Number field is eliminated. This new interpretation enables an ARI Device to support up to 256 Functions [0..255] instead of 8 Functions [0..7].

ARI is controlled by a new set of optional capability and control register bits. These provide:

- Software the ability to detect whether a component supports ARI.
- Software the ability to configure an ARI Downstream Port so the logic that determines when to turn a Type 1 Configuration Request into a Type 0 Configuration Request no longer enforces a restriction on the traditional Device Number field being 0.

- Software the ability to configure an ARI Device to assign each Function to a Function Group. Controls based on Function Groups may be preferable when finer granularity controls based on individual Functions are not required.
 - If Multi-Function VC arbitration is supported and enabled, arbitration can optionally be based on Function Groups instead of individual Functions.
 - If ACS P2P Egress Controls are supported and enabled, access control can optionally be based on Function Groups instead of individual Functions.

The following illustrates an example flow for enabling these capabilities and provides additional details on their usage:

1. Software enumerates the PCI Express hierarchy and determines whether the ARI capability is supported.
 a. For an ARI Downstream Port, the capability is communicated through the Device Capabilities 2 register.

 b. For an ARI Device, the capability is communicated through the ARI Capability structure.

 c. ARI has no impact on the base enumeration algorithms used in platforms today.
2. Software enables ARI functionality in each component.
 a. In an ARI Downstream Port immediately above an ARI Device, software sets the ARI Forwarding Enable bit in the Device Control 2 register. Setting this bit ensures the logic that determines when to turn a Type 1 Configuration Request into a Type 0 Configuration Request no longer enforces a restriction on the traditional Device Number field being 0.

 b. In an ARI Device, Extended Functions are always implicitly enabled. However, once ARI-aware software enables ARI Forwarding in the Downstream Port immediately above the ARI Device, ARI-aware software must discover and configure the Extended Functions.

 c. If an ARI Device implements a Multi-Function VC Capability structure with Function arbitration, and also implements MFVC Function Groups, ARI-aware software categorizes Functions into Function Groups.
 i. Each Function is assigned to a Function Group represented by a Function Group Number.
 ii. A maximum of 8 Function Groups can be configured.
 iii. Within the Multi-Function VC Arbitration Table, a Function Group Number is used in place of a Function Number in each arbitration slot.
 1. Arbitration occurs on a Function Group basis instead of an individual Function basis.
 2. All other aspects of Multi-Function VC arbitration remain unchanged. See Section 7.18.10 for additional details.
 iv. Function arbitration within each Function Group is implementation-specific.

d. If an ARI Device supports ACS P2P Egress Control, access control can be optionally implemented on a Function Group basis.

e. To improve the enumeration performance and create a more deterministic solution, software can enumerate Functions through a linked list of Function Numbers. The next linked list element is communicated through each Function's ARI Capability register.

 i. Function 0 acts as the head of a linked list of Function Numbers. Software detects a non-zero Next Function Number field within the ARI Capability register as the next Function within the linked list. Software issues a configuration probe using the Bus Number captured by the Device and the Function Number derived from the ARI Capability register to locate the next associated Function's configuration space.

 ii. Function Numbers may be sparse and non-sequential in their consumption by an ARI Device.

With an ARI Device, the Phantom Functions Supported field within each Function's Device Capabilities register (see Section 7.8.3, Table 7-13) must be set to 00b to indicate that Phantom Functions are not supported. The Extended Tag Field Enable bit (see Section 7.8.4, Table 7-14) can still be used to enable each Function to support up to 256 outstanding Requests.

Figure 6-13 shows an example system topology with two ARI Devices, one below a Root Port and one below a Switch. For access to Extended Functions in ARI Device X, Root Port A must support ARI Forwarding and have it enabled by software. For access to Extended Functions in ARI Device Y, Switch Downstream Port D must support ARI Forwarding and have it enabled by software. With this configuration, it is recommended that software not enable ARI Forwarding in Root Port B or Switch Downstream Port C.

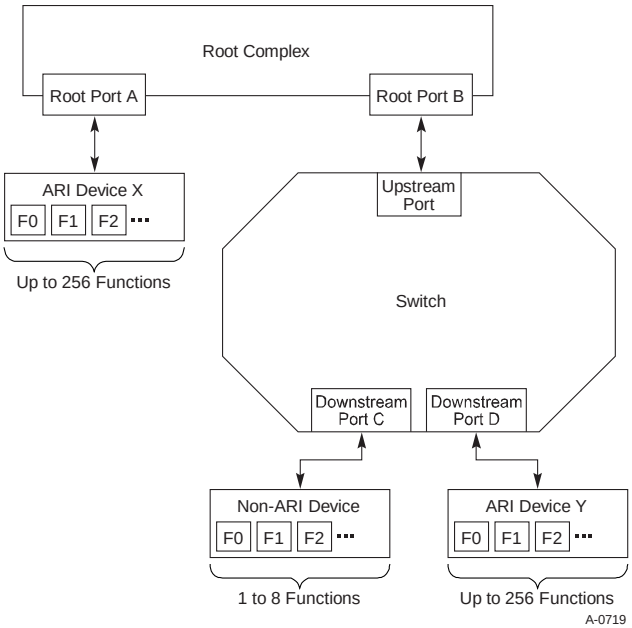


Figure 6-13: Example System Topology with ARI Devices



IMPLEMENTATION NOTE

ARI Forwarding Enable Being Set Inappropriately

It is strongly recommended that software in general Set the ARI Forwarding Enable bit in a Downstream Port only if software is certain that the device immediately below the Downstream Port is an ARI Device. If the bit is Set when a non-ARI Device is present, the non-ARI Device can respond to Configuration Space accesses under what it interprets as being different Device Numbers, and its Functions can be aliased under multiple Device Numbers, generally leading to undesired behavior.

Following a hot-plug event below a Downstream Port, it is strongly recommended that software Clear the ARI Forwarding Enable bit in the Downstream Port until software determines that a newly added component is in fact an ARI Device.



IMPLEMENTATION NOTE

ARI Forwarding Enable Setting at Firmware/OS Control Handoff

It is strongly recommended that firmware not have the ARI Forwarding Enable bit Set in a Downstream Port upon control handoff to an operating system unless firmware knows that the operating system is ARI-aware. With this bit Set, a non-ARI-aware operating system might be able to discover and enumerate Extended Functions in an ARI Device below the Downstream Port, but such an operating system would generally not be able to manage Extended Functions successfully, since it would interpret there being multiple Devices below the Downstream Port instead of a single ARI Device. As one example of many envisioned problems, the interrupt binding for INTx virtual wires would not be consistent with what the non-ARI-aware operating system would expect.

6.14. Multicast Operations

The Multicast Capability structure defines a Multicast address range, the segmentation of that range into a number, N, of equal sized Multicast Windows, and the association of each Multicast Window with a Multicast Group, MCG. Each Function that supports Multicast within a component implements a Multicast Capability structure that provides routing directions and permission checking for each MCG for TLPs passing through or to the Function. The Multicast Group is a field of up to 6 bits in width embedded in the address beginning at the MC_Index_Position, as defined in Section 7.21.4.

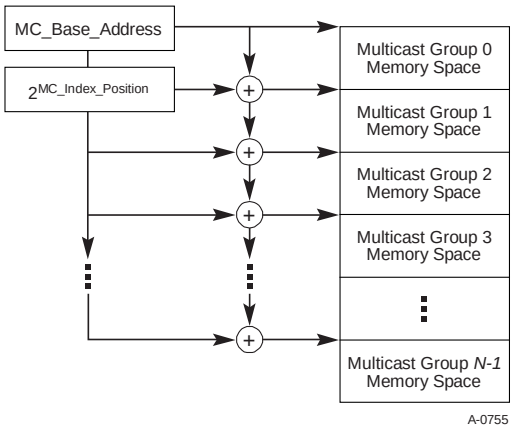


Figure 6-14: Segmentation of the Multicast Address Range

6.14.1. Multicast TLP Processing

A Multicast Hit occurs if all of the following are true:

- MC_Enable is Set
- TLP is a Memory Write or an Address Routed Message, both of which are Posted Requests
- $Address_{TLP}$ >= MC_Base_Address
- $Address_{TLP}$ < (MC_Base_Address+ ($2^{MC_Index_Position} * (MC_Num_Group + 1)$))

In this step, each Switch Ingress Port and other components use values of MC_Enable, MC_Base_Address, MC_Index_Position, and MC_Num_Group from any one of their Functions. Software is required to configure all Functions of a Switch and all Functions of a Multi-Function Upstream Port to have the same values in each of these fields and results are indeterminate if this is not the case.

If the address in a Non-Posted Memory Request hits in a Multicast Window, no Multicast Hit occurs and the TLP is processed normally per the base specification – i.e., as a unicast.

If a Multicast Hit occurs, the only ACS access control that can still apply is ACS Source Validation. In particular, neither ACS redirection nor the ACS Egress Control vector affects operations during a Multicast Hit.

If a Multicast Hit occurs, normal address routing rules do not apply. Instead, the TLP is processed as follows:

The Multicast Group is extracted from the address in the TLP using any Function's values for MC_Base_Address and MC_Index_Position. Specifically:

$$MCG = ((Address_{TLP} - MC_Base_Address) >> MC_Index_Position)\ \&\ 3Fh$$

In this process, the component may use any Function's values for MC_Base_Address and MC_Index_Position. Which Function's values are used is device-specific.

Components next check the MC_Block_All and the MC_Block_Untranslated bits corresponding to the extracted MCG. Switches and Root Ports check Multicast TLPs in their Ingress Ports using the MC_Block_All and MC_Block_Untranslated registers associated with the Ingress Port. Endpoint Functions check Multicast TLPs they are preparing to send, using their MC_Block_All and MC_Block_Untranslated registers. If the MC_Block_All bit corresponding to the extracted MCG is set, the TLP is handled as an MC Blocked TLP. If the MC_Block_Untranslated bit corresponding to the extracted MCG is set and the TLP contains an Untranslated Address, the TLP, is also handled as an MC Blocked TLP.



IMPLEMENTATION NOTE

MC_Block_Untranslated and PIO Writes

Programmed I/O (PIO) Writes to Memory Space generally have Untranslated addresses since there is no architected mechanism for software to control the Address Type (AT) field for PIO Requests. Thus, if it's necessary for a given Switch to Multicast any PIO Writes, software should ensure that the appropriate MC_Block_Untranslated bits in the Upstream Port of that Switch are Clear. Otherwise, the Switch Upstream Port may block PIO Writes that legitimately target Multicast Windows. Since it may be necessary for software to clear MC_Block_Untranslated bits in a Switch Upstream Port for the sake of PIO Writes, the following are strongly recommended for a Root Complex capable of Address translation:·

- All Integrated Endpoints each implement a Multicast Capability structure to provide access control for sending Untranslated Multicast TLPs.·
- All peer-to-peer capable Root Ports each implement a Multicast Capability structure to provide access control for Untranslated Multicast TLPs that are forwarded peer-to-peer.

For similar reasons, with Multicast-capable Switch components where the Upstream Port is a Function in a multi-Function device, it is strongly recommended that any Endpoints in that multi-Function device each implement a Multicast Capability structure.

IMPLEMENTATION NOTE

Multicast Window Size

Each ultimate Receiver of a Multicast TLP may have a different Multicast Window size requirement. At one extreme, a Multicast Window may be required to cover a range of memory implemented within the device. At the other, it may only need to cover a particular offset at which a FIFO register is located. The MC_Window_Size_Requested field within the Multicast Capability register is used by an Endpoint to advertise the size of Multicast Window that it requires.

Unless available address space is limited, resource allocation software may be able to treat each request as a minimum and set the Multicast Window size via MC_Index_Position to accommodate the largest request. In some cases, a request for a larger window size can be satisfied by configuring a smaller window size and assigning the same membership to multiple contiguous MCGs.

IMPLEMENTATION NOTE

Multicast, ATS, and Redirection

The ACS P2P Request Redirection and ACS Direct Translated P2P mechanisms provide a means where P2P Requests with Untranslated Addresses can be redirected to the Root Complex (RC) for access control checking, whereas P2P Requests with Translated Addresses can be routed "directly" to their P2P targets for improved performance. No corresponding redirection mechanism exists for Multicast TLPs.

To achieve similar functionality, an RC might be configured to provide one or more target Memory Space ranges that are not in the Multicast address range, but the RC maps to "protected" Multicast Windows. Multicast TLP senders either with or without ATS capability then target these RC Memory Space ranges in order to access the protected Multicast Windows indirectly. When either type of sender targets these ranges with Memory Writes, each TLP that satisfies the access control checks will be reflected back down by the RC with a Translated Address targeting a protected Multicast Window.[100] ATS-capable senders can request and cache Translated Addresses using the RC Memory Space range, and then later use those Translated Addresses for Memory Writes that target protected Multicast Windows directly and can be Multicast without a taking a trip through the RC.

For hardware enforcement that only Translated Addresses can be used to target the protected Multicast Windows directly, software Sets appropriate MCG bits in the MC_Block_Untranslated register in all applicable Functions throughout the platform. Each MCG whose bit is set will cause its associated Multicast Window to be protected from direct access using Untranslated Addresses.

If the TLP is not blocked in a Switch or Root Complex it is forwarded out all of the Ports, except its Ingress Port, whose MC_Receive bit corresponding to the extracted MCG is set. In an Endpoint, it is consumed by all Functions whose MC_Receive bit corresponding to the extracted MCG is set. If no Ports forward the TLP or no Functions consume it, the TLP is silently dropped.

[100] If the original sender belongs to the MCG associated with this Window, the original sender will also receive a copy of the reflected TLP.

To prevent loops, it is prohibited for a Root Port or a Switch Port to forward a TLP back out its Ingress Port, even if so specified by the MC_Receive register associated with the Port. An exception is the case described in the preceding Implementation Note, where an RC reflects a unicast TLP that came in on an Ingress Root Port to a Multicast Window. In that case, when specified by the MC_Receive register associated with that Ingress Root Port, the RC is required to send the reflected TLP out the same Root Port that it originally came in.

A Multicast Hit suspends normal address routing, including default Upstream routing in Switches. When a Multicast Hit occurs, the TLP will be forwarded out only those Egress Ports whose MC_Receive bit associated with the MCG extracted from the address in the TLP is set. If the address in the TLP does not decode to any Downstream Port using normal address decode, the TLP will be copied to the Upstream Port only if so specified by the Upstream Port's MC_Receive register.

6.14.2. Multicast Ordering

No new ordering rules are defined for processing Multicast TLPs. All Multicast TLPs are Posted Requests and follow Posted Request ordering rules. Multicast TLPs are ordered per normal ordering rules relative to other TLPs in a component's ingress stream through the point of replication. Once copied into an egress stream, a Multicast TLP follows the same ordering as other Posted Requests in the stream.

6.14.3. Multicast Capability Structure Field Updates

Some fields of the Multicast Capability structure may be changed at any time. Others cannot be changed with predictable results unless the MC_Enable bit is Clear in every Function of the component. The latter group includes MC_Base_Address and MC_Index_Position.

Fields which software may change at any time include MC_Enable, MC_Num_Group, MC_Receive, MC_Block_All, and MC_Block_Untranslated. Updates to these fields must themselves be ordered. Consider, for example, TLPs A and B arriving in that order at the same Ingress Port and in the same TC. If A uses value X for one of these fields, then B must use the same value or a newer value.

For Multi-Function Upstream Switch Ports Multicast TLPs received by one Switch or transmitted by one Endpoint Function are presented to the other parallel Endpoint Functions and the Downstream Switch Ports of the other parallel Switches (Functions are considered to be parallel if they are in the same Device. A single Multicast TLP is forwarded Upstream when any of the Upstream Switch Functions has the appropriate MC_Receive bit Set.

6.14.4. MC Blocked TLP Processing

When a TLP is blocked by the MC_Block_All or the MC_Block_Untranslated mechanisms, the TLP is dropped. The Function blocking the TLP serves as the Completer. The Completer must log and signal this MC Blocked TLP error as indicated in Figure 6-2. In addition, the Completer must set the Signaled Target Abort bit in either its Status register or Secondary Status register as appropriate. To assist in fault isolation and root cause analysis, it is highly recommended that AER be implemented in Functions with Multicast capability.

In Root Complexes and Switches, if the error occurs with a TLP received by an Ingress Port, the error is reported by that Ingress Port. If the error occurs in an Endpoint Function preparing to send the TLP, the error is reported by that Endpoint Function.

6.14.5. MC_Overlay Mechanism

The MC Overlay mechanism is provided to allow a single BAR in an Endpoint that doesn't contain a Multicast Capability structure to be used for both Multicast and unicast TLP reception. Software can configure the MC_Overlay mechanism to affect this by setting the MC_Overlay_BAR in a Downstream Port so that the Multicast address range, or a portion of it, is remapped (overlaid) onto the Memory Space range accepted by the Endpoint's BAR. At the Upstream Port of a Switch, the mechanism can be used to overlay a portion of the Multicast address range onto a Memory Space range associated with host memory.

A Downstream Port's MC_Overlay mechanism applies to TLPs exiting that Port. An Upstream Port's MC_Overlay mechanism applies to TLPs exiting the Switch heading Upstream. A Port's MC_Overlay mechanism does not apply to TLPs received by the Port, to TLPs targeting memory space within the Port, or to TLPs routed Peer-to-Peer between Functions in a Multi-Function Upstream Port.

When enabled, the overlay operation specifies that bits in the address in the Multicast TLP, whose bit numbers are equal to or higher than the MC_Overlay_Size field, be replaced by the corresponding bits in the MC_Overlay_BAR. In other words:

```
If (MC_Overlay_Size < 6)
Then Egress_TLP_Addr = Ingress_TLP_Addr;
Else Egress_TLP_Addr = {MC_Overlay_BAR[63:MC_Overlay_Size],
                        Ingress_TLP_Addr[MC_Overlay_Size-1:0]};
```

If the TLP with modified address contains the optional ECRC, the unmodified ECRC will almost certainly indicate an error. The action to be taken if a TLP containing an ECRC is Multicast copied to an Egress Port that has MC_Overlay enabled depends upon whether or not optional support for ECRC regeneration is implemented. All of the contingent actions are outlined in Table 6-10. If MC_Overlay is not enabled, the TLP is forwarded unmodified. If MC_Overlay is enabled and the TLP has no ECRC, the modified TLP, with its address replaced as specified in the previous paragraph is forwarded. If the TLP has an ECRC but ECRC regeneration is not supported, then the modified TLP is forwarded with its ECRC dropped and the TD bit in the header cleared to indicate no ECRC attached. If the TLP has an ECRC and ECRC regeneration is supported, then an ECRC check is performed before the TLP is forwarded. If the ECRC check passes, the TLP is forwarded with regenerated ECRC. If the ECRC check fails, the TLP is forwarded with inverted regenerated ECRC.

Table 6-10: ECRC Rules for MC_Overlay

MC_Overlay Enabled	TLP has ECRC	ECRC Regeneration Supported	Action if ECRC Check Passes	Action if ECRC Check Fails
No	x	x	Forward TLP unmodified	
Yes	No	x	Forward modified TLP	
Yes	Yes	No	Forward modified TLP with ECRC dropped and TD bit clear	
Yes	Yes	Yes	Forward modified TLP with regenerated ECRC	Forward modified TLP with inverted regenerated ECRC



IMPLEMENTATION NOTE

MC_Overlay and ECRC Regeneration

Switch and Root Complex Ports have the option to support ECRC regeneration. If ECRC regeneration is supported, then it is highly advised to do so robustly by minimizing the time between checking the ECRC of the original TLP and replacing it with an ECRC computed on the modified TLP. The TLP is unprotected during this time, leaving a data integrity hole if the pre-check and regeneration aren't accomplished in the same pipeline stage.

Stripping the ECRC from Multicast TLPs passing through a Port that has MC_Overlay enabled but doesn't support ECRC regeneration allows the receiving Endpoint to enable ECRC checking. In such a case, the Endpoint will enjoy the benefits of ECRC on non-Multicast TLPs without detecting ECRC on Multicast TLPs modified by the MC_Overlay mechanism.

When Multicast ECRC regeneration is supported, and an ECRC error is detected prior to TLP modification, then inverting the regenerated ECRC ensures that the ECRC error isn't masked by the regeneration process.



IMPLEMENTATION NOTE

Multicast to Endpoints That Don't Have Multicast Capability

An Endpoint Function that doesn't contain a Multicast Capability structure cannot distinguish Multicast TLPs from unicast TLPs. It is possible for a system designer to take advantage of this fact to employ such Endpoints as Multicast targets. The primary requirement for doing so is that the base and limit registers of the virtual PCI to PCI Bridge in the Switch Port above the device be configured to overlap at least part of the Multicast address range or that the MC_Overlay mechanism be employed. Extending this reasoning, it is even possible that a single Multicast target Function could be located on the PCI/PCI-X side of a PCI Express to PCI/PCI-X Bridge.

If an Endpoint without a Multicast Capability structure is being used as a Multicast target and the MC_Overlay mechanism isn't used, then it may be necessary to read from the Endpoint's Memory Space using the same addresses used for Multicast TLPs. Therefore, Memory Reads that hit in a Multicast Window aren't necessarily errors. Memory Reads that hit in a Multicast Window and that don't also hit in the aperture of a Root Complex Integrated Endpoint or the Downstream Port of a Switch will be routed Upstream, per standard address routing rules, and be handled as a UR there.



IMPLEMENTATION NOTE

Multicast in a Root Complex

A Root Complex with multiple Root Ports that supports Multicast may implement as many Multicast Capability structures as its implementation requires. If it implements more than one, software should ensure that certain fields, as specified in Section 6.14.3, are configured identically. To support Multicast to Root Complex Integrated Endpoints, the implementation needs to expose all TLPs identified as Multicast via the MC_Base_Address register to all potential Multicast target Endpoints integrated within it. Each such Integrated Endpoint then uses the MC_Receive register in its Multicast Capability structure to determine if it should receive the TLP.



IMPLEMENTATION NOTE

Multicast and Multi-Function Devices

All Port Functions and Endpoint Functions that are potential Multicast targets need to implement a Multicast Capability structure so that each has its own MC_Receive vector. Within a single component, software should configure the MC_Enable, MC_Base_Address, MC_Index_Position, and MC_Num_Group fields of these Capability structures identically. That being the case, it is sufficient to implement address decoding logic on only one instance of the Multicast BAR in the component.



IMPLEMENTATION NOTE

Congestion Avoidance

The use of Multicast increases the output link utilization of Switches to a degree proportional to both the size of the Multicast groups used and the fraction of Multicast traffic to total traffic. This results in an increased risk of congestion and congestion spreading when Multicast is used.

To mitigate this risk, components that are intended to serve as Multicast targets should be designed to consume Multicast TLPs at wire speed. Components that are intended to serve as Multicast sources should consider adding a rate limiting mechanism.

In many applications, the application's Multicast data flow will have an inherent rate limit and can be accommodated without causing congestion. Others will require an explicit mechanism to limit the injection rate, selection of a Switch with buffers adequate to hold the requisite bursts of Multicast traffic without asserting flow control, or selection of Multicast target components capable of sinking the Multicast traffic at the required rate. It is the responsibility of the system designer to choose the appropriate mechanisms and components to serve the application.



IMPLEMENTATION NOTE

The Host as a Multicast Recipient

For general-purpose systems, it is anticipated that the Multicast address range will usually not be configured to overlap with Memory Space that's directly mapped to host memory. If host memory is to be included as a Multicast recipient, the Root Complex may need to have some sort of I/O Memory Management Unit (IOMMU) that is capable of remapping portions of Multicast Windows to host memory, perhaps with page-level granularity. Alternatively, the MC_Overlay mechanism in the Upstream Port of a Switch can be used to overlay a portion of the Multicast address range onto host memory.

For embedded systems that lack an IOMMU, it may be feasible to configure Multicast Windows overlapping with Memory Space that's directly mapped to host memory, thus avoiding the need for an IOMMU. Specific details of this approach are beyond the scope of this specification.

6.15. Atomic Operations (AtomicOps)

An Atomic Operation (AtomicOp) is a single PCI Express transaction that targets a location in Memory Space, reads the location's value, potentially writes a new value back to the location, and returns the original value. This "read-modify-write" sequence to the location is performed atomically. AtomicOps include the following:

- FetchAdd (Fetch and Add): Request contains a single operand, the "add" value
 - Read the value of the target location.
 - Add the "add" value to it using two's complement arithmetic ignoring any carry or overflow.
 - Write the sum back to the target location.
 - Return the original value of the target location.
- Swap (Unconditional Swap): Request contains a single operand, the "swap" value
 - Read the value of the target location.
 - Write the "swap" value back to the target location.
 - Return the original value of the target location.
- CAS (Compare and Swap): Request contains two operands, a "compare" value and a "swap" value
 - Read the value of the target location.
 - Compare that value to the "compare" value.
 - If equal, write the "swap" value back to the target location.
 - Return the original value of the target location.

A given AtomicOp transaction has an associated operand size, and the same size is used for the target location accesses and the returned value. FetchAdd and Swap support operand sizes of 32 and 64 bits. CAS supports operand sizes of 32, 64, and 128 bits.

AtomicOp capabilities are optional normative. Endpoints and Root Ports are permitted to implement AtomicOp Requester capabilities. PCI Express Functions with Memory Space BARs as well as all Root Ports are permitted to implement AtomicOp Completer capabilities. Routing elements (Switches, as well as Root Complexes supporting peer-to-peer access between Root Ports) require AtomicOp routing capability in order to route AtomicOp Requests. AtomicOps are architected for device-to-host, device-to-device, and host-to-device transactions. In each case, the Requester, Completer, and all intermediate routing elements must support the associated AtomicOp capabilities.

AtomicOp capabilities are not supported on PCI Express to PCI/PCI-X Bridges. If need be, Locked Transactions can be used for devices below such Bridges. AtomicOps and Locked Transactions can operate concurrently on the same hierarchy.

Software discovers specific AtomicOp Completer capabilities via three new bits in the Device Capabilities 2 register (see Section 7.8.15). For increased interoperability, Root Ports are required to

implement certain AtomicOp Completer capabilities in sets if at all (see Section 6.15.3.1). Software discovers AtomicOp routing capability via the AtomicOp Routing Supported bit in the Device Capabilities 2 register. Software discovery of AtomicOp Requester capabilities is outside the scope of this specification, but software must set the AtomicOp Requester Enable bit in a Function's Device Control 2 register before the Function can initiate AtomicOp Requests (see Section 7.8.16).

With routing elements, software can set an AtomicOp Egress Blocking bit (see Section 7.8.16) on a Port-by-Port basis to avoid AtomicOp Requests being forwarded to components that shouldn't receive them, and might handle each as a Malformed TLP, which by default is a Fatal Error. Each blocked Request is handled as an AtomicOp Egress Blocked error, which by default is an Advisory Non-Fatal Error.

AtomicOps are Memory Transactions, so existing standard mechanisms for managing Memory Space access (e.g., Bus Master Enable, Memory Space Enable, and Base Address registers) apply.

Comment [VS215]: Per Larry Tesdall in email to Steve G.

6.15.1. AtomicOp Use Models and Benefits

AtomicOps enable advanced synchronization mechanisms that are particularly useful when there are multiple producers and/or multiple consumers that need to be synchronized in a non-blocking fashion. For example, multiple producers can safely enqueue to a common queue without any explicit locking.

AtomicOps also enable lock-free statistics counters, for example where a device can atomically increment a counter, and host software can atomically read and clear the counter.

Direct support for the three chosen AtomicOps over PCI Express enables easier migration of existing high-performance SMP applications to systems that use PCI Express as the interconnect to tightly-coupled accelerators, co-processors, or GP-GPUs. For example, a ported application that uses PCI Express-attached accelerators may be able to use the same synchronization algorithms and data structures as the earlier SMP application.

An AtomicOp to a given target generally incurs latency comparable to a Memory Read to the same target. Within a single hierarchy, multiple AtomicOps can be "in flight" concurrently. AtomicOps generally create negligible disruption to other PCI Express traffic.

Compared to Locked Transactions, AtomicOps provide lower latency, higher scalability, advanced synchronization algorithms, and dramatically less impact to other PCI Express traffic.

6.15.2. AtomicOp Transaction Protocol Summary

Detailed protocol rules and requirements for AtomicOps are distributed throughout the rest of this specification, but here is a brief summary plus some unique requirements.

- AtomicOps are Non-Posted Memory Transactions, supporting 32- and 64-bit address formats.
- FetchAdd, Swap, and CAS each use a distinct type code.
- The Completer infers the operand size from the Length field value and type code in the AtomicOp Request.

- The endian format used by AtomicOp Completers to read and write data at the target location is implementation specific, and permitted to be whatever the Completer determines to be appropriate for the target memory (e.g., little-endian, big-endian, etc). See Section 2.2.2.
- If an AtomicOp Requester supports Address Translation Services (ATS), the Requester is permitted to use a Translated address in an AtomicOp Request only if the Translated address has appropriate access permissions. Specifically, the Read (R) and Write (W) fields must both be Set, and the Untranslated access only (U) field must be Clear. See Section 2.2.4.1.
- If a component supporting Access Control Services (ACS) supports AtomicOp routing or AtomicOp Requester capability, it handles AtomicOp Requests and Completions the same as with other Memory Requests and Completions with respect to ACS functionality.
- The No Snoop attribute is applicable and permitted to be Set with AtomicOp Requests, but atomicity must be guaranteed regardless of the No Snoop attribute value.
- The Relaxed Ordering attribute is applicable and permitted to be Set with AtomicOp Requests, where it affects the ordering of both the Requests and their associated Completions.
- Ordering requirements for AtomicOp Requests are similar to those for Non-Posted Write Requests. Thus, if a Requester wants to ensure that an AtomicOp Request is observed by the Completer before a subsequent Posted or Non-Posted Request, the Requester must wait for the AtomicOp Completion before issuing the subsequent Request.
- Ordering requirements for AtomicOp Completions are similar to those for Read Completions.
- Unless there's a higher precedence error, a Completer must handle a Poisoned AtomicOp Request as a Poisoned TLP Received error, and must also return a Completion with a Completion Status of Unsupported Request (UR). See Section 2.7.2.2. The value of the target location must remain unchanged.
- If the Completer of an AtomicOp Request encounters an uncorrectable error accessing the target location or carrying out the Atomic operation, the Completer must handle it as a Completer Abort (CA). The subsequent state of the target location is implementation specific.
- Completers are required to handle any properly formed AtomicOp Requests with types or operand sizes they don't support as an Unsupported Request (UR). If the Length field in an AtomicOp Request contains an unarchitected value, the Request must be handled as a Malformed TLP. See Section 2.2.7.
- If any Function in a multi-Function device supports AtomicOp Completer or AtomicOp routing capability, all Functions with Memory Space BARs in that device must decode properly formed AtomicOp Requests and handle any they don't support as an Unsupported Request (UR). Note that in such devices, Functions lacking AtomicOp Completer capability are forbidden to handle properly formed AtomicOp Requests as Malformed TLPs.
- If an RC has any Root Ports that support AtomicOp routing capability, all Root Complex Integrated Endpoints in the RC reachable by forwarded AtomicOp Requests must decode properly formed AtomicOp Requests and handle any they don't support as an Unsupported Request (UR).
- With an AtomicOp Request having a supported type and operand size, the Completer is required either to carry out the Request or handle it as Completer Abort (CA) for any location in its target Memory Space. Completers are permitted to support AtomicOp Requests on a subset

of their target Memory Space as needed by their programming model (see Section 2.3.1). Memory Space structures defined or inherited by PCI Express (e.g., the MSI-X Table structure) are not required to be supported as AtomicOp targets unless explicitly stated in the description of the structure.

- For a Switch or an RC, when AtomicOp Egress Blocking is enabled in an Egress Port, and an AtomicOp Request targets going out that Egress Port, the Egress Port must handle the Request as an AtomicOp Egress Blocked error[101] (see Figure 6-2) and must also return a Completion with a Completion Status of UR. If the severity of the AtomicOp Egress Blocked error is non-fatal, this case must be handled as an Advisory Non-Fatal Error as described in Section 6.2.3.2.4.1.

6.15.3. Root Complex Support for AtomicOps

RCs have unique requirements and considerations with respect to AtomicOp capabilities.

6.15.3.1. Root Ports with AtomicOp Completer Capabilities

AtomicOp Completer capability for a Root Port indicates that the Root Port supports receiving at its Ingress Port AtomicOp Requests that target host memory or Memory Space allocated by a Root Port BAR. This is independent of any Root Complex Integrated Endpoints that have AtomicOp Completer capabilities.

If a Root Port implements any AtomicOp Completer capability for host memory access, it must implement all 32-bit and 64-bit AtomicOp Completer capabilities. Implementing 128-bit CAS Completer capability is optional.

If an RC has one or more Root Ports that implement AtomicOp Completer capability, the RC must ensure that host memory accesses to a target location on behalf of a given AtomicOp Request are performed atomically with respect to each host processor or device access to that target location range.

If a host processor supports atomic operations via its instruction set architecture, the RC must also ensure that host memory accesses on behalf of a given AtomicOp Request preserve the atomicity of any host processor atomic operations.

6.15.3.2. Root Ports with AtomicOp Routing Capability

As with other PCI Express Transactions, the support for peer-to-peer routing of AtomicOp Requests and Completions between Root Ports is optional and implementation dependent. If an RC supports AtomicOp routing capability between two or more Root Ports, it must indicate that capability in each associated Root Port via the AtomicOp Routing Supported bit in the Device Capabilities 2 register.

An RC is not required to support AtomicOp routing between all pairs of Root Ports that have the AtomicOp Routing Supported bit Set. An AtomicOp Request that would require routing between

[101] Though an AtomicOp Egress Blocked error is handled by returning a Completion with UR Status, the error is not otherwise handled as an Unsupported Request. For example, it does not set the Unsupported Request Detected bit in the Device Status register.

unsupported pairs of Root Ports must be handled as an Unsupported Request (UR), and reported by the "sending" Port.

The AtomicOp Routing Supported bit must be Set for any Root Port that supports forwarding of AtomicOp Requests initiated by host software or Root Complex Integrated Endpoints. The AtomicOp Routing Supported bit must be Set for any Root Ports that support forwarding of AtomicOp Requests received on their Ingress Port to Root Complex Integrated Endpoints.

6.15.3.3. RCs with AtomicOp Requester Capabilities

An RC is permitted to implement the capability for either host software or Root Complex Integrated Endpoints to initiate AtomicOp Requests.

Software discovery of AtomicOp Requester capabilities is outside the scope of this specification.

If an RC supports software-initiated AtomicOp Requester capabilities, the specific mechanisms for how software running on a host processor causes the RC to generate AtomicOp Requests is outside the scope of this specification.



IMPLEMENTATION NOTE

Generating AtomicOp Requests via Host Processor Software

If a host processor instruction set architecture (ISA) supports atomic operation instructions that directly correspond to one or more PCI Express AtomicOps, an RC might process the associated internal atomic transaction that targets PCI Express Memory Space much like it processes the internal read transaction resulting from a processor load instruction. However, instead of "exporting" the internal read transaction as a PCI Express Memory Read Request, the RC would export the internal atomic transaction as a PCI Express AtomicOp Request. Even if an RC uses the "export" approach for some AtomicOp types and operand sizes, it would not need to use this approach for all.

For AtomicOp types and operand sizes where the RC does not use the "export" approach, the RC might use an RC register-based mechanism similar to one where some PCI host bridges use CONFIG_ADDRESS and CONFIG_DATA registers to generate Configuration Requests. Refer to the *PCI Local Bus Specification* for details.

The "export" approach may permit a large number of concurrent AtomicOp Requests without becoming RC register limited. It may also be easier to support AtomicOp Request generation from user space software using this approach.

The RC register-based mechanism offers the advantage of working for all AtomicOp types and operand sizes even if the host processor ISA doesn't support the corresponding atomic instructions. It might also support a polling mode for waiting on AtomicOp Completions as opposed to stalling the processor while waiting for a Completion.

6.15.4. Switch Support for AtomicOps

If a Switch supports AtomicOp routing capability for any of its Ports, it must do so for all of them.

6.16. Dynamic Power Allocation (DPA) Capability

A common approach to managing power consumption is through a negotiation between the device driver, operating system, and executing applications. Adding Dynamic Power Allocation for such devices is anticipated to be done as an extension of that negotiation, through software mechanisms that are outside of the scope of this specification. Some devices do not have a device specific driver to manage power efficiently. The DPA Capability provides a mechanism to allocate power dynamically for these types of devices. DPA is optional normative functionality applicable to Endpoint Functions that can benefit from the dynamic allocation of power and do not have an alternative mechanism.

The DPA Capability enables software to actively manage and optimize Function power usage when in the D0 state. DPA is not applicable to power states D1-D3 therefore the DPA Capability is independently managed from the PCI-PM Capability.

DPA defines a set of power substates, each of which with an associated power allocation. Up to 32 substates [0..31] can be defined per Function. Substate 0, the default substate, indicates the maximum power the Function is ever capable of consuming.

Substates must be contiguously numbered from 0 to Substate_Max, as defined in Section 7.24.2. Each successive substate has a power allocation lower than or equal to that of the prior substate. For example, a Function with four substates could be defined as follows:

1. Substate 0 (the default) defines a power allocation of 25 Watts.
2. Substate 1 defines a power allocation of 20 Watts.
3. Substate 2 defines a power allocation of 20 Watts.
4. Substate 3 defines a power allocation of 10 Watts.

When the Function is initialized, it will operate within the power allocation associated with substate 0. Software is not required to progress through intermediate substates. Over time, software may dynamically configure the Function to operate at any of the substates in any sequence it chooses. Software is permitted to configure the Function to operate at any of the substates before the Function completes a previously initiated substate transition.

On the completion of the substate transition(s) the Function must compare its substate with the configured substate. If the Function substate does not match the configured substate, then the Function must begin transition to the configured substate. It is permitted for the Function to dynamically alter substate transitions on Configuration Requests instructing the Function to operate in a new substate.

In the prior example, software can configure the Function to transition to substate 4, followed by substate 1, followed by substate 3, and so forth. As a result, the Function must be able to transition between any substates when software configures the associated control field.

The Substate Control Enabled bit provides a mechanism that allows the DPA Capability to be used in conjunction with the software negotiation mechanism mentioned above. When Set, power allocation is controlled by the DPA Capability. When Clear, the DPA Capability is disabled, and the Function is not permitted to directly initiate substate transitions based on configuration of the Substate Control register field. At an appropriate point in time, software participating in the

software negotiation mechanism mentioned above clears the bit, effectively taking over control of power allocation for the Function.

It is required that the Function respond to Configuration Space accesses while in any substate.

At any instant, the Function must never draw more power than it indicates through its Substate Status. When the Function is configured to transition from a higher power substate to a lower power substate, the Function's Substate Status must indicate the higher power substate during the transition, and must indicate the lower power substate after completing the transition. When the Function is configured to transition from a lower power substate to a higher power substate, the Function's Substate Status must indicate the higher power substate during the transition, as well as after completing the transition.

Due to the variety of applications and the wide range of maximum power required for a given Function, the transition time required between any substates is implementation specific. To enable software to construct power management policies (outside the scope of this specification), the Function defines two Transition Latency Values. Each of the Function substates associates its maximum Transition Latency with one of the Transition Latency Values, where the maximum Transition Latency is the time it takes for the Function to enter the configured substate from any other substate. A Function is permitted to complete the substate transition faster than the maximum Transition Latency for the substate.

6.16.1. DPA Capability with Multi-Function Devices

It is permitted for some or all Functions of a multi-Function device to implement a DPA Capability. The power allocation for the multi-Function device is the sum of power allocations set by the DPA Capability for each Function. It is permitted for the DPA Capability of a Function to include the power allocation for the Function itself as well as account for power allocation for other Functions that do not implement a DPA Capability. The association between multiple Functions for DPA is implementation specific and beyond the scope of this specification.

6.17. TLP Processing Hints (TPH)

TLP Processing Hints is an optional feature that provides hints in Request TLP headers to facilitate optimized processing of Requests that target Memory Space. These Processing Hints enable the system hardware (e.g., the Root Complex and/or Endpoints) to optimize platform resources such as system and memory interconnect on a per TLP basis. The TPH mechanism defines Processing Hints that provide information about the communication models between Endpoints and the Root-complex. Steering Tags are system-specific values used to identify a processing resource that a Requester explicitly targets. System software discovers and identifies TPH capabilities to determine the Steering Tag allocation for each Function that supports TPH.

6.17.1. Processing Hints

The Requester provides hints to the Root Complex or other targets about the intended use of data and data structures by the host and/or device. The hints are provided by the Requester, which has knowledge of upcoming Request patterns, and which the Completer would not be able to deduce autonomously (with good accuracy). Cases of interest to distinguish with such hints include:

DWHR: Device writes then host reads soon

HWDR: Device reads data that the Host is believed to have recently written

D*D*: Device writes/reads, then device reads/writes soon

Includes DWDW, DWDR, DRDW, DRDR

Bi-Directional: Data structure that is shared and has equal read/write access by host and device.

The usage models are mapped to the Processing Hint encodings as described in Table 6-11.

Table 6-11: Processing Hint Mapping

PH[1:0] (b)	Processing Hint	Usage Model
00	Bi-directional data structure	Bi-Directional shared data structure
01	Requester	D*D*
10	Target	DWHR HWDR
11	Target with Priority	Same as target but with temporal re-use priority

6.17.2. Steering Tags

Functions that intend to target a TLP towards a specific processing resource such as a host processor or system cache hierarchy require topological information of the target cache (e.g., which host cache). Steering Tags are system-specific values that provide information about the host or cache structure in the system cache hierarchy. These values are used to associate processing elements within the platform with the processing of Requests.

Software programmable Steering Tag values to be used are stored in an ST Table that is permitted to be located either in the TPH Requester Capability structure (see Section 7.26) or combined with the MSI-X Table (see Section 7.7), but not in both locations for a given Function. When the ST Table is combined with the MSI-X Table, the 2 most significant bytes of the Vector Control register of each MSI-X Table entry are used to contain the Steering Tag value.

The choice of ST Table location is implementation specific and is discoverable by software. A Function that implements MSI-X is permitted to locate the ST Table in either location (see Section 7.26.2). A Function that implements both MSI and MSI-X is permitted to combine the ST Table with the MSI-X Table and use it, even when MSI-X is disabled (i.e. when MSI is enabled). Each ST Table entry is 2 bytes. The size of the ST Table is indicated in the TPH Requester Capability structure.

Comment [VS216]: Errata A10

For some usage models the Steering Tags are not required or not provided, and in such cases a Function is permitted to use a value of all zeroes in the ST field to indicate no ST preference. The association of each Request with an ST Table entry is device specific and outside the scope of this specification.

6.17.3. ST Modes of Operation

The ST Table Location field in the TPH Requester Capability structure indicates where (if at all) the ST Table is implemented by the Function. If an ST Table is implemented, software can program it with the system-specific Steering Tag values.

Table 6-12: ST Modes of Operation

ST Mode Select [2:0] (b)	ST Mode Name	Description
000	No ST Mode	The Function must use a value of all zeroes for all Steering Tags.
001	Interrupt Vector Mode	Each Steering Tag is selected by an MSI/MSI-X interrupt vector number. The Function is required to use the Steering Tag value from an ST Table entry that can be indexed by a valid MSI/MSI-X interrupt vector number.
010	Device Specific Mode	It is recommended for the Function to use a Steering Tag value from an ST Table entry, but it is not required.
All other encodings	Reserved	Reserved for future use.

In the No ST Mode of operation, the Function must use a value of all zeroes for each Steering Tag, enabling the use of Processing Hints without software-provided Steering Tags.

In the Interrupt Vector Mode of operation, Steering Tags are selected from the ST Table using MSI/MSI-X interrupt vector numbers. For Functions that have MSI enabled, the Function is required to select tags within the range specified by the Multiple Message Enable field in the MSI Capability structure. For Functions that have MSI-X enabled, the Function is required to select tags within the range of the MSI-X Table size. If the ST Table Size is smaller than the enabled range of interrupt vector numbers, the Function is permitted to either not use TPH for certain transactions, to use TPH with a Steering Tag of 0 or to use TPH with an implementation defined mechanism used to select a Steering Tag value from the ST Table. If the ST Table Size is larger than the enabled range of interrupt vector numbers, ST Table Entries corresponding to out of range interrupt vector numbers are ignored by the Function.

Comment [VS217]: Errata A9

In the Device Specific Mode of operation, the assignment of the Steering Tags to Requests is device specific. The number of Steering Tags used by the Function is permitted to be different than the number of interrupt vectors allocated for the Function, irrespective of the ST Table location, and Steering Tag values used in Requests are not required to come from the architected ST Table.

A Function that is capable of generating TPH Requests is required to support the No ST Mode of operation. Support for other ST Modes of operation is optional. Only one ST Mode of operation can be selected at a time by programming ST Mode Select.



IMPLEMENTATION NOTE

ST Table Programming

To ensure that deterministic Steering Tag values are used in Requests, it is recommended that software either quiesce the Function or disable the TPH Requester capability during the process of performing ST Table updates. Failure to do so may result in non-deterministic values of ST values being used during ST Table updates.

6.17.4. TPH Capability

TPH capabilities are optional normative. Each Function capable of generating Request TLPs with TPH is required to implement a TPH Requester Capability structure. Functions that support processing of TLPs with TPH as Completers are required to indicate TPH Completer capability via the Device Capabilities 2 register. TPH is architected to be applied for transactions that target Memory Space, and is applicable for transaction flows between device-to-host, device-to-device and host-to-device. In each case for TPH to be supported, the Requester, Completer, and all intermediate routing elements must support the associated TPH capabilities.

Software discovers the Requester capabilities via the TPH Requester Capability structure and Completer capabilities via the Device Capabilities 2 register (see Section 7.8.15). Software must program the TPH Requester Enable field in the TPH Requester Capability structure to enable the Function to initiate Requests with TPH.

TPH only provides additional information to enable optimized processing of Requests that target Memory Space, so existing mechanisms and rules for managing Memory Space access (e.g., Bus Master Enable, Memory Space Enable, and Base Address registers) are not altered.

6.18. Latency Tolerance Reporting (LTR) Mechanism

The Latency Tolerance Reporting (LTR) mechanism enables Endpoints to report their service latency requirements for Memory Reads and Writes to the Root Complex, so that power management policies for central platform resources (such as main memory, RC internal interconnects, and snoop resources) can be implemented to consider Endpoint service requirements. The LTR Mechanism does not directly affect Link power management or Switch internal power management, although it is possible that indirect effects will occur.

The implications of "latency tolerance" will vary significantly between different device types and implementations. When implementing this mechanism, it will generally be desirable to consider if service latencies impact functionality or only performance, if performance impacts are linear, and how much it is possible for the device to use buffering and/or other techniques to compensate for latency sensitivities.

The Root Complex is not required to honor the requested service latencies, but is strongly encouraged to provide a worst case service latency that does not exceed the latencies indicated by the LTR mechanism.

LTR support is discovered and enabled through reporting and control registers described in Chapter 7. Software must not enable LTR in an Endpoint unless the Root Complex and all intermediate Switches indicate support for LTR. Note that it is not required that all Endpoints support LTR to permit enabling LTR in those Endpoints that do support it. When enabling the LTR mechanism in a hierarchy, devices closest to the Root Port must be enabled first.

If an LTR Message is received at a Downstream Port that does not support LTR or if LTR is not enabled, the Message must be treated as an Unsupported Request.

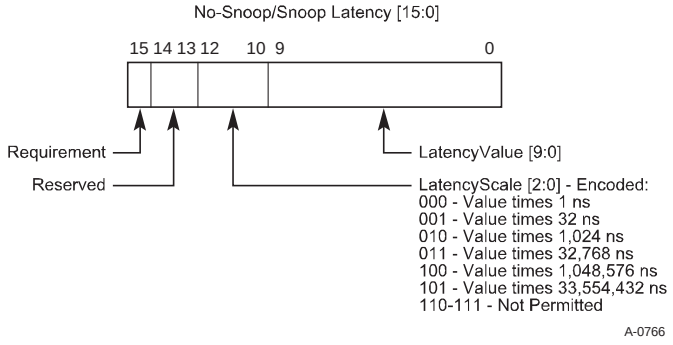


Figure 6-15: Latency Fields Format for LTR Messages

No-Snoop Latency and Snoop Latency: As shown in Figure 6-15, these fields include a Requirement bit that indicates if the device has a latency requirement for the given type of Request. If the Requirement bit is Set, the LatencyValue and LatencyScale fields describe the latency requirement. If the Requirement bit is Clear, there is no latency requirement and the LatencyValue and LatencyScale fields are ignored. With any LTR Message transmission, it is permitted for a device

Comment [VS218]: Errata B25

to indicate that a requirement is being reported for only no-snoop Requests, for only snoop Requests, or for both types of Requests. It is also permitted for a device to indicate that it has no requirement for either type of traffic, which it does by clearing the Requirement bit in both fields.

Each field also includes value and scale fields that encode the reported latency. Values are multiplied by the indicated scale to yield an absolute time value, expressible in a range from 1 ns to $2^{25}*(2^{10}-1) = 34{,}326{,}183{,}936$ ns.

Setting the value and scale fields to all 0's indicates that the device will be impacted by any delay and that the best possible service is requested.

If a device doesn't implement or has no service requirements for a particular type of traffic, then it must have the Requirement bit clear for the associated latency field.

When directed to a non-D0 state by a Write to the PMCSR register, if a device's most recently transmitted LTR message (since the last DL_Down to DL_Up transition) reported one or both latency fields with any Requirement bit set, then it must send a new LTR Message with both of the Requirement bits clear prior to transitioning to the non-D0 state.

When the LTR Mechanism Enable bit is cleared, if a device's most recently sent LTR Message (since the last DL_DOWN to DL_Up transition) reported latency tolerance values with any Requirement bit set, then it must send a new LTR Message with all the Requirement bits clear.

An LTR Message from a device reflects the tolerable latency from the perspective of the device, for which the platform must consider the service latency itself, plus the delay added by the use of Clock Power Management (CLKREQ#), if applicable. The service latency itself is defined as follows:

- When a ~~If the~~ device issues a ~~Read~~ Non-Posted Request, service latency of that Request is ~~measured as~~ the delay from transmission of the last ~~END~~ symbol of~~in~~ the Request TLP to the receipt of the first~~STP~~ symbol of~~in~~ the first Completion TLP for that Request[102].
- When a ~~If the~~ device issues one or more ~~Write~~ Posted Requests such that it cannot issue another ~~Write~~ Posted Request due to Flow Control backpressure, ~~the~~ service latency of the blocked Request is ~~measured from~~ the delay from the transmission of the ~~END~~ last symbol of the previous Posted Request~~TLP that exhausts the FC credits~~ to the receipt of the ~~SDP~~ first symbol of the DLLP returning ~~more credits for that Request type~~the credit(s) that allows transmission of the blocked Request[100].

If Clock Power Management is used, then the platform implementation-dependent period between when a device asserts CLKREQ# and the device receives a valid clock signal constitutes an additional component of the platform service latency that must be comprehended by the platform when setting platform power management policy.

It is recommended that Endpoints transmit an LTR Message Upstream shortly after LTR is enabled.

It is strongly recommended that Endpoints send no more than two LTR Messages within any 500 µs time period, except where required to by the specification. Downstream Ports must not generate an error if more than two LTR Messages are received within a 500 µs time period, and must properly handle all LTR messages regardless of the time interval between them.

Comment [VS219]: Errata B26

Comment [VS220]: Errata B26

Formatted: Font: (Default) Arial, Superscript

[102] For this definition, all of the symbols of a DLLP or TLP are included. For 8b10b, the first and last symbols are framing symbols (SDP, STP or END, see Section 4.2.1). For 128b/130b, the first symbol of a packet is the first symbol of a framing token (SDP or STP) and the last symbol of a packet is the last symbol of a CRC or LCRC (see Section 4.2.2).

Multi-Function devices (MFDs) associated with an Upstream Port must transmit a "conglomerated" LTR Message Upstream according to the following rules:

- The acceptable latency values for the Message sent Upstream by the MFD must reflect the lowest values associated with any Function.
 - It is permitted that the snoop and no-snoop latencies reported in the conglomerated Message are associated with different Functions.
 - If none of the Functions report a requirement for a certain type of traffic (snoop/no-snoop), the Message sent by the MFD must not set the Requirement bit corresponding to that type of traffic.
- The MFD must transmit a new LTR Message Upstream when any Function of the MFD changes the values it has reported internally in such a way as to change the conglomerated value earlier reported by the MFD.

Switches must collect the Messages from Downstream Ports that have the LTR mechanism enabled and transmit a "conglomerated" Message Upstream according to the following rules:

- If a Switch supports the LTR feature, it must support the feature on its Upstream Port and all Downstream Ports.
- A Switch Upstream Port is permitted to transmit LTR Messages only when its LTR Mechanism Enable bit is Set or shortly after software clears its LTR Mechanism Enable bit as described earlier in this section.
- The acceptable latency values for the Message sent Upstream by the Switch must be calculated as follows:
 - If none of the Downstream Ports receive an LTR Message containing a requirement for a certain type of traffic (snoop/no-snoop), then any LTR Message sent by the Switch must not set the Requirement bit corresponding to that type of traffic.
 - Define LTRdnport[N] as the value reported in the LTR Message received at Downstream Port N, with these adjustments if applicable:
 - LTRdnport[N] is effectively infinite if the Requirement bit is clear or if a Not Permitted LatencyScale value is used
 - LTRdnport[N] must be 0 if the Requirement bit is 1 and the LatencyValue field is all 0's regardless of the LatencyScale value
 - Define LTRdnportMin as the minimum value of LTRdnport[N] across all Downstream Ports
 - Define Lswitch as all latency induced by the Switch
 - If Lswitch dynamically changes based on the Switch's operational mode, the Switch must not allow Lswitch to exceed 20% of LTRdnportMin, unless Lswitch for the Switch's lowest latency mode is greater, in which case the lowest latency state must be used
 - Calculate the value to transmit upstream, LTRconglomerated, as LTRdnportMin – Lswitch, unless this value is less than 0 in which case LTRconglomerated is 0

- If LTRconglomerated is 0, both the LatencyValue and LatencyScale fields must be all 0's in the conglomerated LTR message

- When a Switch Downstream Port goes to DL_Down status, the previous latencies recorded for that Port must be treated as invalid. A new LTR message must be transmitted Upstream if the conglomerated latencies are changed as a result.
- If a Switch Downstream Port has the LTR Mechanism Enable bit cleared, the Latency Tolerance values recorded for that Port must be treated as invalid, and the latencies to be transmitted Upstream updated and a new conglomerated Message transmitted Upstream if the conglomerated latencies are changed as a result.
- A Switch must transmit an LTR Message Upstream when any Downstream Port/Function changes the latencies it has reported in such a way as to change the conglomerated latency reported by the Switch.
- A Switch must not transmit LTR Messages Upstream unless triggered to do so by one of the events described above.

The RC is permitted to delay processing of device Request TLPs provided it satisfies the device's service requirements.

When the latency requirement is updated during a series of Requests, it is required that the updated latency figure be comprehended by the RC prior to servicing a subsequent Request. In all cases the updated latency value must take effect within a time period equal to or less than the previously reported latency requirement. It is permitted for the RC to comprehend the updated latency figure earlier than this limit.



IMPLEMENTATION NOTE

Optimal Use of LTR

It is recommended that Endpoints transmit an updated LTR Message each time the Endpoint's service requirements change. If the latency tolerance is being reduced, it is recommended to transmit the updated LTR Message ahead of the first anticipated Request with the new requirement, allowing the amount of time indicated in the previously issued LTR Message. If the tolerance is being increased, then the update should immediately follow the final Request with the preceding latency tolerance value.

Typically, the Link will be in ASPM L1, and, if Clock Power Management (Clock PM) is supported, CLKREQ# will be deasserted, at the time an Endpoint reaches an internal trigger that causes the Endpoint to initiate Requests to the RC. The following text shows an example of how LTR is applied in such a case. Key time points are illustrated in Figure 6-16.

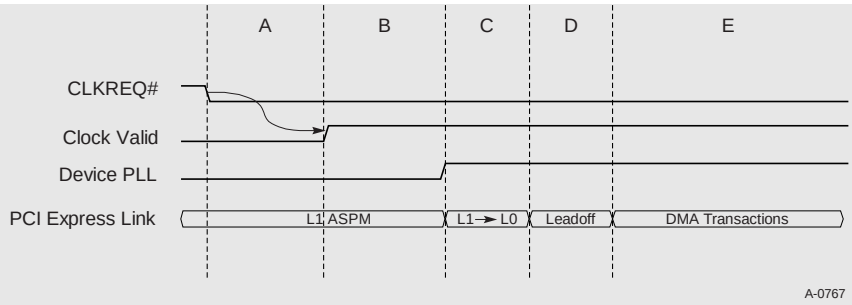


Figure 6-16: CLKREQ# and Clock Power Management

Time A is a platform implementation-dependent period between when a device asserts CLKREQ# and the device receives a valid clock signal. This value will not exceed the latency in effect.

Time B is the device implementation-dependent period between when a device has a valid clock and it can initiate the retraining sequence to transition from L1 ASPM to L0.

Time C is the period during which the transition from L1 ASPM to L0 takes place

Time D for a Read transaction is the time between the transmission of the END symbol in the Request TLP to the receipt of the STP symbol in the Completion TLP for that Request. Time D for a Write transaction is the time between the transmission of the END symbol of the TLP that exhausts the FC credits to the receipt of the SDP symbol in the DLLP returning more credits for that Request type. This value will not exceed the latency in effect.

Time E is the period where the data path from the Endpoint to system memory is open, and data transactions are not subject to the leadoff latency.

The LTR latency semantic reflects the tolerable latency seen by the device as measured by one or both of the following:

Case 1: the device may or may not support Clock PM, but has not deasserted its CLKREQ# signal – The latency observed by the device is represented in Figure 6-16 as the sum of times C and D.

Case 2: the device supports Clock PM and has deasserted CLKREQ#- The latency observed by the device is represented as the sum of times A, C, and D.

To effectively use the LTR mechanism in conjunction with Clock PM, the device will know or be able to measure times B and C, so that it knows when to assert CLKREQ#. The actual values of Time A, Time C, and Time D may vary dynamically, and it is the responsibility of the platform to ensure the sum will not exceed the latency.

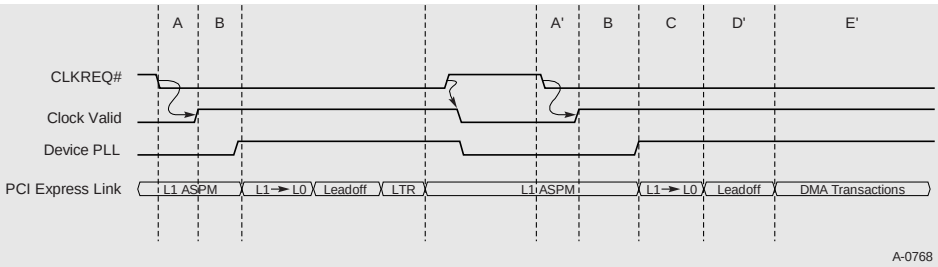


Figure 6-17: Use of LTR and Clock Power Management

In a very simple model, an Endpoint may choose to implement LTR as shown in Figure 6-17. When an Endpoint determines that it is idle, it sends an LTR Message with the software configured maximum latency or the maximum latency the Endpoint can support.

When the Endpoint determines that it has a need to maintain sustained data transfers with the Root Complex, the Endpoint sends a new LTR Message with a shorter latency (at Time E). This LTR Message is sent prior to the next data flush by a time equal to the maximum latency sent before (the time between Time E and Time D'). In between Time E and Time A', the Endpoint can return to a low power state, while the platform transitions to a state where it can provide the shorter latency when the device next needs to transmit data.

Note that the RC may delay processing of device Request TLPs, provided it satisfies the device's service requirements. If, for example, an Endpoint connected to Root Port 1 reports a latency tolerance of 100 µs, and an Endpoint on Root Port 2 report a value of 30 µs, the RC might implement a policy of stalling an initial Request following an idle period from Root Port 1 for 70 µs before servicing the Request with a 30 µs latency, thus providing a perceived service latency to the first Endpoint of 100 µs. This RC behavior provides the RC the ability to batch together Requests for more efficient servicing.

It is recommended that Endpoints buffer Requests as much as possible, and then use the full Link bandwidth in bursts that are as long as the Endpoint can practically support, as this will generally lead to the best overall platform power efficiency.

Note that LTR may be enabled in environments where not all Endpoints support LTR, and in such environments, Endpoints that do not support LTR may experience suboptimal service.

6.19. Optimized Buffer Flush/Fill (OBFF) Mechanism

The Optimized Buffer Flush/Fill (OBFF) Mechanism enables a Root Complex to report to Endpoints (throughout a hierarchy) time windows when the incremental platform power cost for Endpoint bus mastering and/or interrupt activity is relatively low. Typically this will correspond to time that the host CPU(s), memory, and other central resources associated with the Root Complex are active to service some other activity, for example the operating system timer tick. The nature and determination of such windows is platform/implementation specific.

An OBFF indication is a hint - Functions are still permitted to initiate bus mastering and/or interrupt traffic whenever enabled to do so, although this will not be optimal for platform power and should be avoided as much as possible.

OBFF is indicated using either of the WAKE# signal or a message (see Section 2.2.8.9). The message is to be used exclusively on interconnects where the WAKE# signal is not available. WAKE# signaling of OBFF or CPU Active must only be initiated by a Root Port when the system is in an operational state, which in an ACPI compliant system corresponds to the S0 state. Functions that are in a non-D0 state must not respond to OBFF or CPU Active signaling.

The OBFF message routing is defined as 100b, for point-to-point, and is only permitted to be transmitted in the Downstream direction. There are multiple OBFF events distinguished. When using the OBFF Message, the OBFF Code field is used to distinguish between different OBFF cases:

- 1111b "CPU Active" – System fully active for all Device actions including bus mastering and interrupts
- 0001b "OBFF" – System memory path available for Device memory read/write bus master activities
- 0000b "Idle" – System in an idle, low power state

All other codes are Reserved.

These codes correspond to various assertion patterns of WAKE# when using WAKE# signaling, as shown in Figure 6-18. There is one negative-going transition when signaling OBFF and two negative going transitions each time CPU Active is signaled. The electrical parameters required when using WAKE# are defined in the WAKE# Signaling section of *PCI Express CEM Specification, Revision 2.0* (or later).

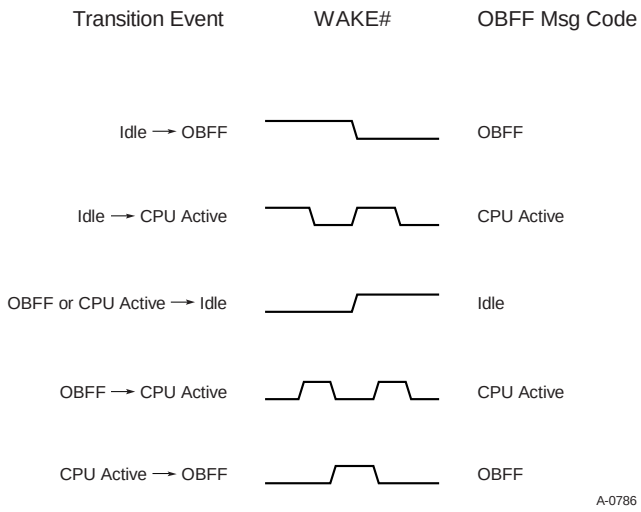


Figure 6-18: Codes and Equivalent WAKE# Patterns

When an OBFF Message is received that indicates a Reserved code, the Receiver, if OBFF is enabled, must treat the indication as a "CPU Active" indication.

An OBFF Message received at a Port that does not implement OBFF or when OBFF is not enabled must be handled as an Unsupported Request (UR).

OBFF indications reflect central resource power management state transitions, and are signaled using WAKE# when this is supported by the platform topology, or using a Message when WAKE# is not available. OBFF support is discovered and enabled through reporting and control registers described in Chapter 7. Software must not enable OBFF in an Endpoint unless the platform supports delivering OBFF indications to that Endpoint.

When the platform indicates the start of a CPU Active or OBFF window, it is recommended that the platform not return to the Idle state in less than 10 µs. It is permitted to indicate a return to Idle in advance of actually entering platform idle, but it is strongly recommended that this only be done to prevent late Endpoint activity from causing an immediate exit from the idle state, and that the advance time be as short as possible.

It is recommended that Endpoints not assume CPU Active or OBFF windows will remain open for any particular length of time.

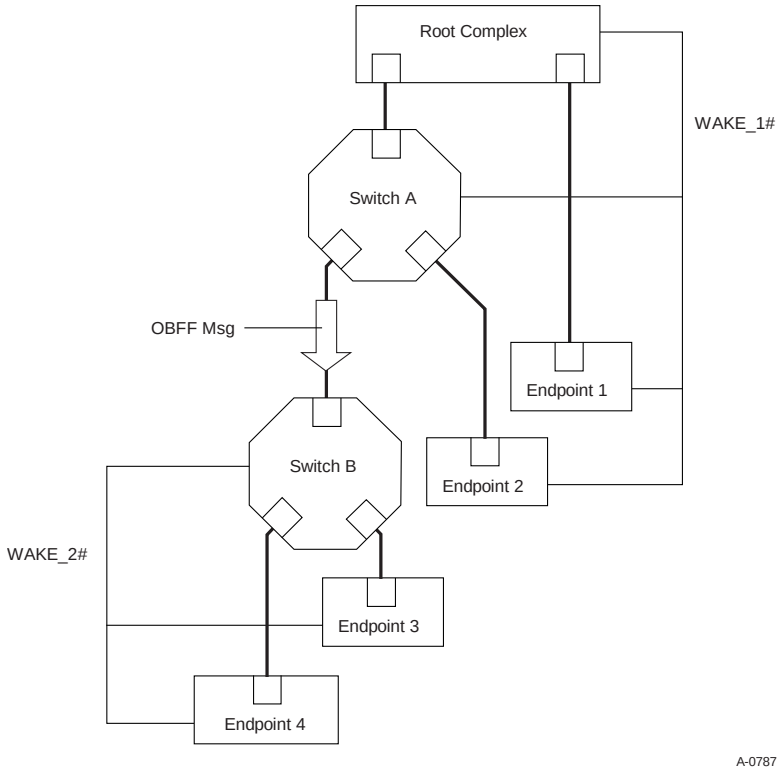


Figure 6-19: Example Platform Topology Showing a Link Where OBFF is Carried by Messages

Figure 6-19 shows an example system where it is necessary for a Switch (A) to translate OBFF indications received using WAKE# into OBFF Messages, which in this case are received by another Switch (B) and translated back to using WAKE# signaling. A HwInit configuration mechanism (set by hardware or firmware) is used to identify cases such as shown in this example (where the link between Switch A and Switch B requires the use of OBFF Messages), and system firmware/software must configure OBFF accordingly.

When a Switch is configured to use OBFF Message signaling at its Upstream Port and WAKE# at one or more Downstream Ports, or vice-versa, when enabled for OBFF, the Switch is required to convert all OBFF indications received at the Upstream Port into the appropriate form at the Downstream Port(s).

When using WAKE#, the enable for any specific Root Port enables the global use of WAKE# unless there are multiple WAKE# signals, in which case only the associated WAKE# signals are affected. When using Message signaling for OBFF, the enable for a particular Root Port enables transmission of OBFF messages from that Root Port only. To ensure OBFF is fully enabled in a platform, all Root Ports indicating OBFF support must be enabled for OBFF. It is permitted for system firmware/software to selectively enable OBFF, but such enabling is beyond the scope of this specification.

To minimize power consumption, system firmware/software is strongly recommended to enable Message signaling of OBFF only when WAKE# signaling is not available for a given link.

OBFF signaling using WAKE# must only be reported as supported by all components connected to a Switch if it is a shared WAKE# signal. In these topologies it is permitted for software to enable OBFF for components connected to the Switch even if the Switch itself does not support OBFF.

It is permitted, although not encouraged, to indicate the same OBFF event more than once in succession.

When a Switch is propagating OBFF indications Downstream, it is strongly encouraged to propagate all OBFF indications. However, especially when using Messages, it may be necessary for the Switch to discard or collapse OBFF indications. It is permitted to discard and replace an earlier indication of a given type when an indication of the same or a different type is received.

Downstream Ports can be configured to transmit OBFF Messages in two ways, which are referred to as Variation A and Variation B. For Variation A, the Port must transmit the OBFF Message if the Link is in the L0 state, but discard the Message when the Link is in the Tx_L0s or L1 state. This variation is preferred when the Downstream Port leads to Devices that are expected to have communication requirements that are not time-critical, and where Devices are expected to signal a non-urgent need for attention by returning the Link state to L0. For Variation B, the Port must transmit the OBFF Message if the Link is in the L0 state, or, if the Link is in the Tx_L0s or L1 state, it must direct the Link to the L0 state and then transmit the OBFF Message. This variation is preferred when the Downstream Port leads to devices that can benefit from timely notification of the platform state.

When initially configured for OBFF operation, the initial assumed indication must be the CPU Active state, regardless of the logical value of the WAKE# signal, until the first transition is observed.

When enabling Ports for OBFF, it is recommended that all Upstream Ports be enabled before Downstream Ports, and Root Ports must be enabled after all other Ports have been enabled. For hot pluggable Ports this sequence will not generally be possible, and it is permissible to enable OBFF using WAKE# to an unconnected hot pluggable Downstream Port. It is recommended that unconnected hot pluggable Downstream Ports not be enabled for OBFF message transmission.



IMPLEMENTATION NOTE

OBFF Considerations for Endpoints

It is possible that during normal circumstances, events could legally occur that could cause an Endpoint to misinterpret transitions from an Idle window to a CPU Active window or OBFF window. For example, a non-OBFF Endpoint could assert WAKE# as a wakeup mechanism, masking the system's transitions of the signal. This could cause the Endpoint to behave in a manner that would be less than optimal for power or performance reasons, but should not be unrecoverable for the Endpoint or the host system.

In order to allow an Endpoint to maintain the most accurate possible view of the host state, it is recommended that the Endpoint place its internal state tracking logic in the CPU Active state when it receives a request that it determines to be host-initiated, and at any point where the Endpoint has a pending interrupt serviced by host software.

Comment [VS221]: ECN: PASID

6.20. PASID TLP Prefix

The PASID TLP Prefix is an End-End TLP Prefix as defined in Section 2.2.1. Layout of the PASID TLP Prefix is shown in Figure 6-20 and Table 6-13.

When a PASID TLP Prefix is present, the PASID value in the prefix, in conjunction with the requester ID, identifies the Process Address Space ID associated with the Request. Each Function has a distinct set of PASID values. PASID values used by one Function are unrelated to PASID values used by any other Function.

A PASID TLP Prefix is permitted on:

- Memory Requests (including AtomicOp Requests) with Untranslated Addresses (See Section 2.2.4.1).
- Translation Requests and Translation Message Requests as defined in the *Address Translation Services Specification*.

The PASID TLP Prefix is not permitted on any other TLP.

6.20.1. Managing PASID TLP Prefix Usage

Usage of PASID TLP Prefixes must be specifically enabled. Unless enabled, a component is not permitted to transmit a PASID TLP Prefix.

For Endpoint Functions (including Root Complex Integrated Devices), the following rules apply:

- A Function is not permitted to send and receive TLPs with a PASID TLP Prefix unless PASID Enable is Set (see Section 7.29.3):
- A Function must have a mechanism for dynamically associating use of a PASID with a particular Function context. This mechanism is device specific.
- A Function must have a mechanism to request that it gracefully stop using a specific PASID. This mechanism is device specific but must satisfy the following rules:
 - A Function may support a limited number of simultaneous PASID stop requests. Software should defer issuing new stop requests until older stop requests have completed.
 - A stop request in one Function must not affect operation of any other Function.
 - A stop request must not affect operation of any other PASID within the Function.
 - A stop request must not affect operation of transactions that are not associated with a PASID.
 - When the stop request mechanism indicates completion, the Function has:
 - Stopped queuing new Requests for this PASID.
 - Completed all Non-Posted Requests associated with this PASID.
 - Flushed to the host all Posted Requests addressing host memory in all TCs that were used by the PASID. The mechanism used for this is device specific (for example: a non-relaxed Posted Write to host memory or a processor read of the

Function can flush TC0; a zero length read to host memory can flush non-zero TCs).

- Optionally flushed all Peer-to-Peer Posted Requests to their destination(s). The mechanism used for this is device specific.
- Complied with additional rules described in the Address Translation Services Specification if Address Translations or Page Requests were issued on the behalf of this PASID.

For Root Complexes, the following rules apply:

- A Root Complex must have a device specific mechanism for indicating support for PASID TLP Prefixes.
- A Root Complex that supports PASID TLP Prefixes must have a device specific mechanism for enabling them. By default usage of PASID TLP Prefixes is disabled.
- A Root Complex that supports PASID TLP Prefixes may optionally have a device specific mechanism for enabling them at a finer granularity than the entire Root Complex (e.g., distinct enables for a specific Root Port, Requester ID, Bus Number, Requester ID, or Requester ID/PASID combination).

6.20.2. PASID TLP Layout

A TLP may contain at most one PASID TLP Prefix.

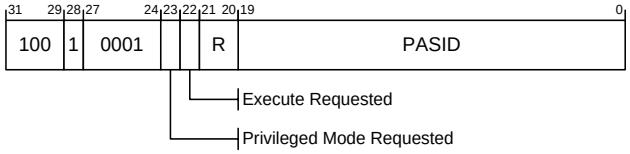


Figure 6-20. PASID TLP Prefix:

Table 6-13. PASID TLP Prefix

Bits	Description
31:29	100b – indicating TLP Prefix
28	1b – indicating End-End TLP Prefix
27:24	0001b – indicating PASID TLP Prefix
23	**Privileged Mode Requested** – If Set indicates a Privileged Mode entity in the Endpoint is issuing the Request, If Clear, indicates a Non-Privileged Mode entity in the Endpoint is issuing the Request. Usage of this bit is specified in Section 6.20.2.3.

Bits	Description
22	**Execute Requested** – If Set indicates the Endpoint is requesting Execute Permission. If Clear, indicates the Endpoint is not requesting Execute Permission. Usage of this bit is specified in Section 6.20.2.2.
21:20	Reserved
19:0	**Process Address Space ID (PASID)** – This field contains the PASID value associated with the TLP. Usage of this field is defined in Section 6.20.2.1.

6.20.2.1. PASID field

Formatted: Font color: Black

The PASID field identifies the user process associated with a Request. This field is present in all PASID TLP Prefixes.

The PASID field is 20 bits wide. Endpoints and Root Complexes need not support the entire range of the field. For Endpoints, the Max PASID Width field indicates the supported range of PASID values (Section 7.28.2). For Root Complexes, an implementation specific mechanism is used to provide this information.

Endpoints are not permitted to send TLPs with a PASID TLP Prefix unless the PASID Enable bit (Section 7.28.3) is Set. Endpoints that support the PASID TLP Prefix must signal Unsupported Request (UR) when they receive a TLP with a PASID TLP Prefix and the PASID Enable bit is Clear.

Root Complexes may optionally support TLPs with PASID TLP Prefixes. The mechanism used to detect whether a Root Complex supports the PASID TLP Prefix is implementation specific.

For Endpoints, the following rules apply:

- The Endpoint is not permitted to send TLPs with a PASID value greater than or equal to $2^{\text{Max PASID Width}}$.
- The Endpoint is optionally permitted to signal an error when it receives a TLP with a PASID value greater than or equal to $2^{\text{Max PASID Width}}$. For Requests, this error is Unsupported Request and for Completions, this error is Unexpected Completion.

For Root Complexes, the following rules apply:

- A Root Complex is not permitted to send a TLP with a PASID value greater than it supports.
- A Root Complex is optionally permitted to signal an error when it receives a TLP with a PASID value greater than it supports. For Requests, this error is Unsupported Request and for Completions, this error is Unexpected Completion.

For Completers, the following rules apply:

- For Untranslated Memory Requests, the PASID value and the Untranslated Address are both used in determining the Translated Address used in satisfying the Request.

For address translation related TLPs, usage of this field is defined in the *Address Translation Services Specification*.

6.20.2.2. Execute Requested

If the Execute Requested bit is Set, the Endpoint is requesting permission for the Endpoint to Execute instructions in the memory range associated with this request. The meaning of Execute permission is outside the scope of this specification.

Endpoints are not permitted to send TLPs with the Execute Requested bit Set unless the Execute Permission Supported bit (Section 7.28.2) and the Execute Permission Enable bit (Section 7.28.3) are both Set.

For Root Complexes, the following rules apply:

- Support for Execute Requested by the Root Complex is optional. The mechanism used to determine whether a Root Complex supports Execute Requested is implementation specific.
- A Root Complex that supports the Execute Requested bit should have an implementation specific mechanism to enable it to use the bit.
- A Root Complex that supports the Execute Requested bit may have an implementation specific mechanism to enable use of the bit at a finer granularity (e.g. for a specific Root Port, for a specific Bus Number, for a specific Requester ID, or for a specific Requester ID/PASID combination), and its default value is implementation specific.

For Completers, the following rules apply:

- Completers have a concept of an effective value of the bit. For a given Request, if the Execute Requested bit is supported and it usage is enabled for the Request, the effective value of the bit is the value in the Request; otherwise the effective value of the bit is 0b.
- For Untranslated Memory Read Requests, Completers use the effective value of the bit as part of the protection check. If this protection check fails, Completers treat the Request as if the memory was not mapped.
- For Untranslated Memory Requests, other than an Untranslated Memory Read Request, the bit is Reserved.

For address translation related TLPs, usage of this bit is defined in the *Address Translation Services Specification*.

6.20.2.3. Privileged Mode Requested

If Privileged Mode Requested is Set, the Endpoint is issuing a Request that targets memory associated with Privileged Mode. If Privileged Mode Requested is Clear, the Endpoint is issuing a Request that targets memory associated with Non-Privileged Mode.

The meaning of Privileged Mode and Non-Privileged Mode and what it means for an Endpoint to be operating in Privileged or Non-Privileged Mode depends on the protection model of the system and is outside the scope of this specification.

Endpoints are not permitted to send a TLP with the Privileged Mode Requested bit Set unless both the Privileged Mode Supported bit (Section 7.28.2) and the Privileged Mode Enable bit (Section 7.28.3) are Set.

For Root Complexes, the following rules apply:

- Support for the Privileged Mode Requested bit by the Root Complex is optional. The mechanism used to determine whether a Root Complex supports the Privileged Mode Requested bit is implementation specific.
- A Root Complex that supports the Privileged Mode Requested bit should have an implementation specific mechanism to enable it to use the bit.
- A Root Complex that supports the Privileged Mode Requested bit may have an implementation specific mechanism to enable use of the bit at a finer granularity (e.g. for a specific Root Port, for a specific Bus Number, for a specific Requester ID, or for a specific Requester ID/PASID combination).

For Completers, the following rules apply:

- Completers have the concept of an effective value of the bit. For a given Request, if the Privileged Mode Requested bit is supported and its usage is enabled for the Request, the effective value of the bit is the value in the Request; otherwise the effective value of the bit is the 0b.
- For Untranslated Memory Requests, Completers use the effective value of the bit as part of its protection check. If this protection check fails, Completers treat the Request as if the memory was not mapped.
- For address translation related TLPs, usage of this bit is defined in the *Address Translation Services Specification*.

6.21. Lightweight Notification (LN) Protocol

Comment [VS222]: ECN: Lightweight Notification (LN) Protocol

Lightweight Notification (LN) protocol enables Endpoints to register interest in specific cachelines in host memory, and be notified via a hardware mechanism when they are updated. An LN Requester (LNR) is a client subsystem in an Endpoint that sends LN Read/Write Requests and receives LN Messages. An LN Completer (LNC) is a service subsystem in the host that receives LN Read/Write Requests, and sends LN Messages when registered cachelines are updated.

LN protocol provides a notification service for when cachelines of interest are updated. Most commonly, an LNR sends an LN Read to a Memory Space range that has an associated LNC, requesting a copy of a cacheline (abbreviated "line" in this section). The LNC returns the requested line to the LNR and records that the LNR has requested notification when that line is updated; that is, the LNC "registers" the line. Later, the LNC notifies the LNR via an LN Message when some entity updates the line, so the LNR can take appropriate action. A similar notification service exists for LN Writes, where an LNR writing a line can be notified later when the line is updated.

LN protocol permits multiple LNRs each to register the same line concurrently. An Endpoint with LNR is permitted to write to a line at any time, regardless of whether the LNR has registered the line.

A typical system consists of host processors, host memory, a host internal fabric, Root Ports, Switches, and Endpoints. Figure 6-21 below illustrates a simplified view of the elements and how they are interconnected to provide a context for describing how LN protocol operates.

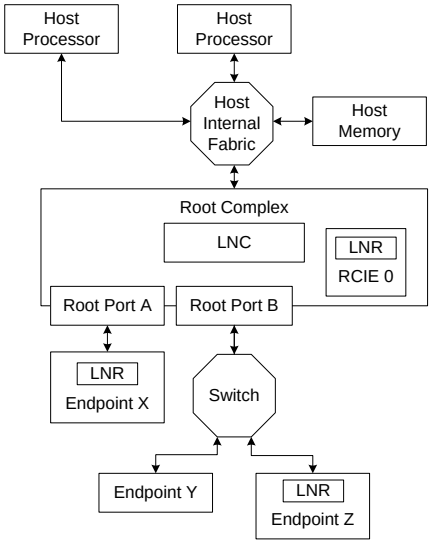


Figure 6-21: Sample System Block Diagram

In the figure above, Endpoint X, Endpoint Z, and RCIE 0 each contain an LNR. The Root Complex contains an LNC.

6.21.1. LN Protocol Operation

LN is a simple protocol that satisfies several key use models with minimum complexity and cost.

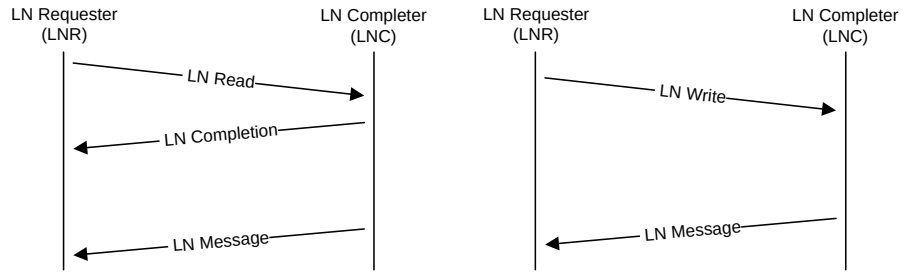


Figure 6-22: LN Protocol Basic Operation

LN protocol operation with a single LNR is shown in the diagram above. For the case of reads, as shown on the left: (1) an LNR requests a copy of a line from host memory using an LN Read; (2) the LNC returns the line in an LN Completion and records that the LNR has registered the line; and (3) the LNC later notifies the LNR using an LN Message when the registered line has been updated. For the case of writes, as shown on the right: (1) an LNR writes to a line to host memory using an LN Write; (2) the LNC records that the LNR has registered the line; and (3) the LNC later notifies the LNR using an LN Message when the registered line has been updated.

An LNC must send an LN Message either if a registered line is updated by some entity (e.g., a CPU or a device) or if the LNC is no longer going to track the state of that line. The latter case is referred to as an eviction, and is indicated by the Notification Reason (NR) field in the LN Message.

If an LN Requester does an LN Read or LN Write to a line that it already has registered, then the LN Requester generally can't determine if a subsequent LN Message it receives for that line was for the most recent LN Read/Write Request or for a previous one.

LN protocol permits multiple LNRs to register the same line concurrently. In this case the LNC notifies the multiple LNRs either by sending a directed LN Message to each LNR, or by sending a broadcast LN Message to each affected Hierarchy Domain.

Comment [VS223]: A20

Once the LNC updates or evicts a given line and sends one or more LN Messages to notify all LNRs with associated registrations, the LNC will not send additional LN Messages for that specific line unless it receives a new LN Read or LN Write to that line.

An LNC is permitted to have an implementation-specific limit on how many LNRs it can independently track for each given line. At or below this limit, the LNC generally uses directed LN Messages for notifications. Above this limit, the LNC uses broadcast LN Messages. An implementation is permitted to have a limit of zero, and use broadcast LN Messages exclusively.

A single LN Message with a specific NR field value can indicate that all lines registered to that LNR have been evicted. Both directed and broadcast versions of this "all evicted" LN Message are permitted.

An LNC is permitted to have an implementation-specific limitations on how many lines or sets of lines it can register concurrently. If the LNC receives a Request to register a new line, but the LNC has insufficient resources to do so, the LNC is permitted to evict one or more old lines in order to free up the necessary resources, or send a LN Message indicating that the new line was evicted.

IMPLEMENTATION NOTE

Excessive Use of Broadcast LN Messages

In order to avoid performance issues, LNCs that use broadcast LN Messages should be implemented to minimize the number of Hierarchy Domains each broadcast LN Message is sent to, and also to keep the rate of broadcast LN Messages within reasonable bounds. Each broadcast LN Message consumes Link bandwidth, and some Endpoints may process broadcast LN Messages at a relatively low rate.

In particular, Endpoints that do not support LN Requester capability may handle received broadcast LN Messages as an exception case using a low performance mechanism, e.g., decoding the Message with device firmware instead of in hardware. Each Message could conceivably take microseconds to process[103], and an excessive rate of them could easily cause back-pressure of Posted Requests within the fabric, causing severe performance problems.

While Endpoints that do not support LN Requester capability may also handle directed LN Messages as an exception case using a low performance mechanism, this should not cause performance problems. With LN protocol, LNCs send directed LN Messages only to Endpoints that support LN Requester capability, and presumably those Endpoints will be able to process both directed and broadcast LN Messages at a rate that avoids performance issues.

6.21.2. LN Registration Management

Since LNCs have limited resources for registering cachelines, an LNR Capability structure in each LNR provides mechanisms for software to limit the LNR's maximum number of outstanding registrations. Software reads the LNR Registration Max field to discover the maximum number an LNR is capable of having outstanding. If necessary, software can set the LNR Registration Limit field to impose a specified limit.

LNC registration resource limitations may be highly implementation specific, perhaps including factors such as set associativity, maximum number of sets, and distinct sets of resources for different regions of host memory. How software discovers these resource limitations is outside the scope of this specification.

To manage its number of outstanding registrations, an LNR can deregister outstanding registrations by sending a zero-length LN Write to each line it wants to deregister.

[103] The Posted Request Acceptance Limit permits an Endpoint to take up to 10 µs to process each received Posted Request. See Section 2.3.1.

Formatted: Font: Arial Black, 14 pt, Font color: Black, All caps

Comment [VS224]: Errata A20

The LNC is not required to accept registrations for all locations in host memory. However, the granularity for memory regions accepting registrations is required to be no finer than aligned 4KB regions. To determine if a given aligned 4KB region accepts registrations, an LNR can perform an LN Read to any cacheline within the region, and see if an LN Completion is returned. If an LNR wishes to "probe" a region for registration capability without actually creating a registration, the LNR can use a zero-length LN Read, which is required to provide this semantic.

6.21.3. LN Ordering Considerations

LN Reads have the same ordering requirements as other Memory Read Requests, and LN Writes have the same ordering requirements as other Memory Write Requests. LN Completions (for LN Reads) have the same ordering requirements as Completions for other Memory Reads. LN Messages have the same ordering requirements as other Posted Requests.

For a given line, when an LN Completer receives an LN Read followed by an update to that line, the LN Completer is permitted to send the LN Completion and the LN Message in either order.

For a given line, when an LN Completer receives an LN Read that triggers an eviction LN Message, the LN Completer is permitted to send the LN Completion and the LN Message in either order.

For a given line, when an LN Completer receives an LN Write that triggers an eviction LN Message, followed by an update to that line, the LN Completer is permitted to send the two LN Messages in either order.

For different lines, an LN Completer is permitted to send LN Messages in any order.

6.21.4. LN Software Configuration

LN protocol supports 2 cacheline sizes (CLSs) – 64 bytes and 128 bytes. Support for each CLS is optional, both for Root Complexes and Endpoints. The CLS in effect for the system is determined by the host, and indicated by the LN System CLS field in applicable Root Ports and RCRBs. All Root Ports and RCRBs that indicate LN Completer support must indicate the same CLS; otherwise, the results are undefined. The host must not change the system CLS while any operating system is running; otherwise, the results are undefined.

Endpoints supporting LN protocol must support one or both CLSs, and indicate this via the LNR-64 Supported and LNR-128 Supported capability bits. When enabling each LN Requester, software must ensure that the associated LNR CLS control bit is configured to match the system CLS; otherwise, the results are undefined. An LN Requester that supports only one CLS is permitted to hardwire its LNR CLS control bit to the corresponding value.

Software must not change the value of the LNR CLS control bit or the LNR Registration Limit field unless its LNR Enable control bit is already Clear or is being Cleared with the same Configuration Write; otherwise, the results are undefined. If the LNR Enable bit is already Clear, software is permitted to change the values of the LNR CLS bit and LNR Registration Limit field concurrently with Setting the LNR Enable bit, using a single Configuration Write.

Software is permitted to Clear the LNR Enable bit at any time. When the LNR Enable bit is Clear, the LNR must clear all its internal registration state.

6.21.5. LN Protocol Summary

Detailed rules and requirements for LN protocol are distributed throughout the rest of this specification, but here is a general summary plus some unique requirements.

- An LN Read is a Memory Read Request whose TLP Header has the LN bit Set. An LN Completion is a Completion whose TLP Header has the LN bit Set. An LN Write is a Memory Write Request whose TLP Header has the LN bit Set.
- All requirements for Memory Read Requests apply to LN Reads unless explicitly stated otherwise.
 - An LN Read must access no more than a single cacheline, as determined by the system CLS. An LN Completer must handle a violation of this rule as a Completer Abort unless the Completer detects a higher precedence error. Partial cacheline LN Reads, including zero-length LN Reads, are permitted.
 - If an LN Completer handles an LN Read as an Uncorrectable Error or an Advisory Non-Fatal Error, the LN Completer must not register notification service for that Request.
 - An LN Completer must handle a zero-length LN Read as a "probe" of the targeted memory region for registration capability. See Section 6.20.2. If the Completion Status is Successful Completion, the LN bit in the TLP Header must indicate if the region supports registration capability, but the LNC must not create a registration for this case. LN Completers must support registration capability with a granularity no finer than aligned 4KB regions.
 - Ordering and Flow Control rules for LN Reads are identical to those for Memory Read Requests.
- All requirements for Memory Read Completions apply to LN Completions unless explicitly stated otherwise.
 - The Completion for an LN Read must be an LN Completion (i.e., have the LN bit Set in its TLP Header) if the Completer is an LN Completer, the targeted memory region accepts registrations, and the Completion Status is Successful Completion; otherwise the Completion must have the LN bit Clear in its TLP Header. Note that a poisoned Completion will have a Completion Status of Successful Completion. See Section 2.7.2.2.
 - Ordering and Flow Control rules for LN Completions are identical to Completions whose TLP Header has the LN bit Clear.
- All requirements for Memory Write Requests apply to LN Writes unless explicitly stated otherwise.
 - An LN Write must access no more than a single cacheline, as determined by the system CLS. An LN Completer must handle a violation of this rule as a Completer Abort unless the Completer detects a higher precedence error.
 - If an LN Completer handles an LN Write as an Uncorrectable Error, the LN Completer must not register notification service for that Request. Note that depending upon its

configuration, a Completer may handle a poisoned LN Write as an Uncorrectable Error, an Advisory Non-Fatal Error, or a masked error.

- An LN Completer must handle a zero-length LN Write as a request to deregister an existing registration. See Section 6.20.2. If the cacheline targeted by a zero-length LN Write was not previously registered, it must remain unregistered.
- Ordering, Flow Control, and Data Poisoning rules for LN Writes are identical to those for Memory Write Requests.
- A Requester must not generate LN Writes for MSI or MSI-X interrupts. An LN Completer must handle an LN Write targeting an interrupt address range as a Completer Abort unless the Completer detects a higher precedence error.

- For LN Reads and LN Writes, the address must be the proper type, as indicated by the Address Type (AT) field. See Section 2.2.4.1. The proper type depends upon whether a Translation Agent (TA) is being used. See the *Address Translation Services (ATS) Specification*.
 - If a TA is being used, the address must be a Translated address. An LN Requester must support ATS in order to acquire and use Translated addresses.
 - If a TA is not being used, the address must be a Default/Untranslated Address.
 - An LN Completer detecting a violation of the above rules must handle the Request as a Completer Abort (CA), unless a higher precedence error is detected.

6.22. Precision Time Measurement (PTM) Mechanism

Comment [VS225]: ECN: Precision Time Measurement

6.22.1. Introduction

Precision Time Measurement (PTM) enables precise coordination of events across multiple components with independent local time clocks. Ordinarily, such precise coordination would be difficult given that individual time clocks have differing notions of the value and rate of change of time. To work around this limitation, PTM enables components to calculate the relationship between their local times and a shared PTM Master Time: an independent time domain associated with a PTM Root.

PTM defines the following:

- PTM Requester – A Function capable of using PTM as a consumer associated with an Endpoint or an Upstream Port.
- PTM Responder – A Function capable of using PTM to supply PTM Master Time associated with a Port or an RCRB.
- Time Source – A local clock associated with a PTM Responder.
- PTM Root – The source of PTM Master Time for a PTM Hierarchy. A PTM Root must also be a Time Source and is typically also a PTM Responder.

Each PTM Root supplies PTM Master Time for a PTM Hierarchy: a set of PTM Requesters associated with a single PTM Root.

Figure 6-23. illustrates some example system topologies using PTM. These are only illustrative examples, and are not intended to imply any limits or requirements.

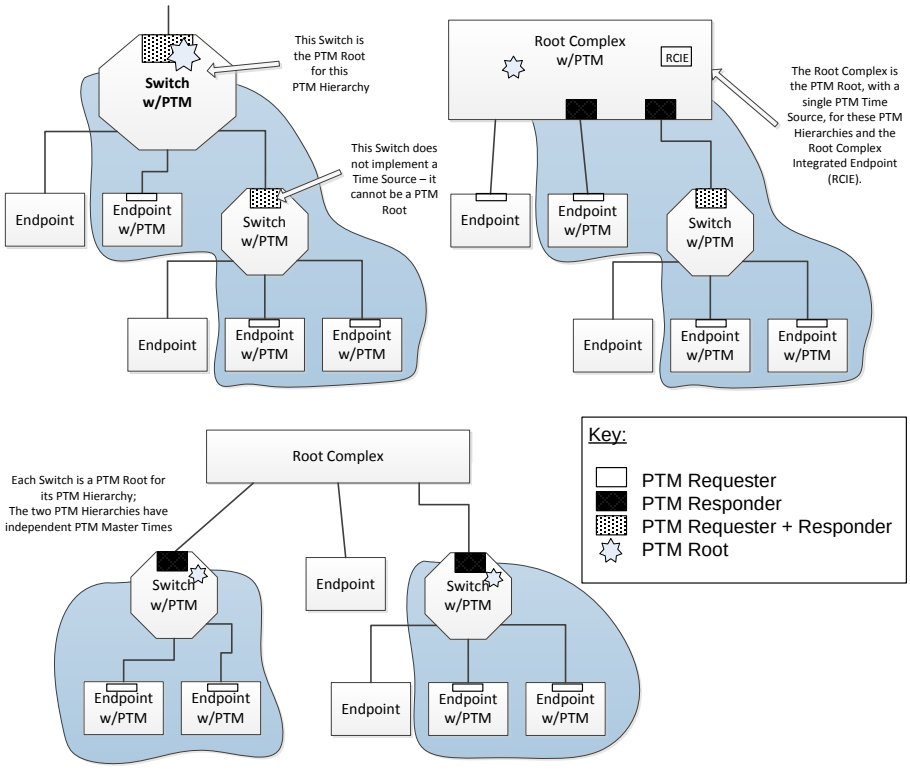


Figure 6-23: Example System Topologies using PTM

6.22.2. PTM Link Protocol

When using PTM between two components on a Link, the Upstream Port, which acts on behalf of the PTM Requester, sends PTM Requests to the Downstream Port on the same Link, which acts on behalf of the PTM Responder.

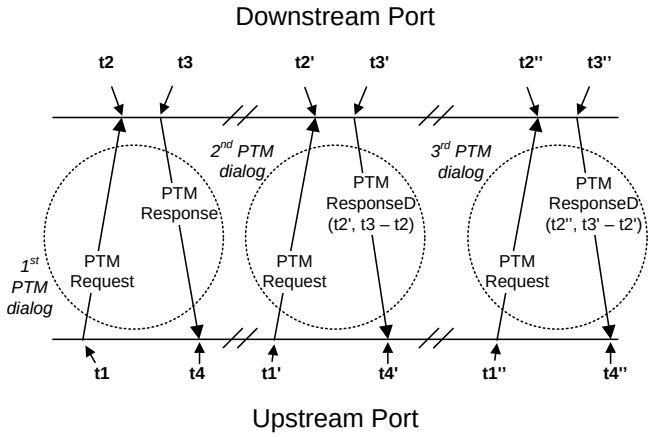


Figure 6-24: Precision Time Measurement Link Protocol

Figure 6-24 illustrates the PTM link protocol. The points t1, t2, t3, and t4 in the above diagram represent timestamps captured locally by each Port as they transmit and receive PTM Messages. The component associated with each Port stores these timestamps from the 1st PTM dialog in internal registers for use in the 2nd PTM dialog, and so on for subsequent PTM dialogs.

The Upstream Port, on behalf of the PTM Requester, initiates the PTM dialog by transmitting a PTM Request message.

The Downstream Port, on behalf of the PTM Responder, has knowledge of or access (directly or indirectly) to the PTM Master Time.

During each dialog, the Downstream Port populates the PTM ResponseD message based on timestamps stored during previous PTM dialogs, as defined in Section 6.22.3.2.

Once each component has historical timestamps from the preceding dialog, the component associated with the Upstream Port can combine its timestamps with those passed in the PTM ResponseD message to calculate the PTM Master Time using the following formula:

$$\text{PTM Master Time at t1}' = t2' - \frac{((t4 - t1) - (t3 - t2))}{2}$$

The values t1, t2, t3, t4, and t2' indicate the timestamps captured during the PTM dialog as illustrated in Figure 6-24.

PTM capable components would typically record the results of these timestamp calculations, and may make them available to software via implementation specific means. Herein, this document refers to this resultant timing information as the component's "PTM context".

For a Switch implementing PTM, the time synchronization mechanism(s) within the Switch itself are implementation-specific.

Formatted: Font color: Black

IMPLEMENTATION NOTE

PTM Theory and Operation

The timestamps captured during the PTM dialogs enable the calculation of the timing relationship between the PTM Requester and PTM Responder. The value (t3-t2) measures the time consumed by the PTM Responder for a given PTM dialog. The time (t4-t1) is the time from request to response. Therefore ((t4 – t1) – (t3 – t2)) effectively gives the round trip message transit time between the two components, and that quantity divided by 2 approximates the Link delay - the time difference between t1 and t2. It is assumed that the Link transit times from PTM Requester to PTM Responder and back again are symmetric, which is typically a good assumption (see also the Implementation Note on PTM Timestamp Capture Mechanisms).

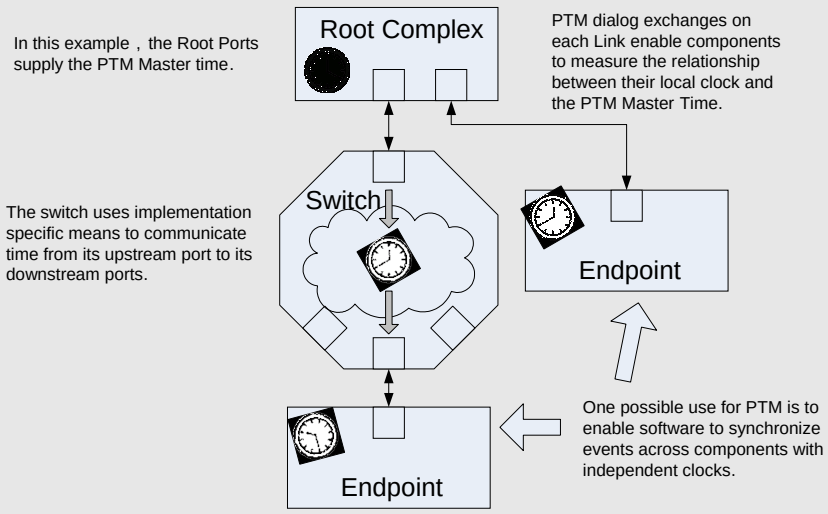


Figure 6-25: Precision Time Measurement Example

Figure 6-25 illustrates a simple device hierarchy employing PTM. Each Upstream Port initiates PTM dialogs to establish the relationship between its local time and the PTM Master Time provided by the Root Port.

In this example, the Switch initiates PTM dialogs on its Upstream Port to obtain the PTM Master Time for use in fulfilling PTM Request Message received at its Downstream Ports. This Switch employs implementation specific means to communicate the PTM Master Time from its Upstream Port to its Downstream Ports.

PTM capable components can make their PTM context available for inspection by software, enabling software to translate timing information between local times and PTM Master Time. In turn, this capability enables software to coordinate events across multiple components with very fine precision.

Similarly, it is strongly recommended that platforms implementing PTM also make the PTM Master Time available to software.

6.22.3. Configuration and Operational Requirements

Software must not have the PTM Enable bit Set in the PTM Control register on a Function associated with an Upstream Port unless the associated Downstream Port on the Link already has the PTM Enable bit Set in its associated PTM Control register.

PTM support by a Function is indicated by the presence of a PTM Extended Capability structure. It is not required that all Endpoints in a hierarchy support PTM, and it is not required for software to enable PTM in all Endpoints that do support it.

If a PTM Message is received by a Port that does not support PTM, or by a Downstream Port when the PTM Enable bit is clear, the Message must be treated as an Unsupported Request. This is a reported error associated with the Receiving Port (see Section 6.2). A Properly formed PTM Response received by an Upstream Ports that support PTM, but for which the PTM Enable bit is clear, must be silently discarded.

As observed through PTM, the PTM Master Time must satisfy the following behavioral requirements:

- Time values must be monotonic, and strictly increasing.
- The perceived granularity must be no greater than the value reported in the Local Clock Granularity field of the PTM Capability register.
- The perceived time must start no later than when the PTM Root processes its first PTM Request Message.

Referring to Figure 6-24, the following rules define timestamp capture:

- A PTM Requester must update its stored t1 timestamp when transmitting a PTM Request Message, even if that transmission is a replay.
- A PTM Responder must update its stored t2 timestamp when receiving a PTM Request Message, even if received TLP is a duplicate.
- A PTM Responder must update its stored t3 timestamp when transmitting a PTM Response or ResponseD Message, even if that transmission is a replay.
- A PTM Requester must update its stored t4 timestamp when receiving a PTM Response Message, even if received TLP is a duplicate.
- Timestamps must be based on the STP Symbol or Token that frames the TLP, as if observing the first bit of that Symbol or Token at the Port's pins.
 - Typically this will require an implementation specific adjustment to compensate for the inability to directly measure the time at the actual pins, as the time will commonly be measured at some internal point in the Rx or Tx path. The accuracy and consistency of this measurement are not bounded by this specification, but it is strongly recommended that the highest practical level of accuracy and consistency be achieved.

Comment [VS226]: Errata A22

6.22.3.1. PTM Requester Role

- Support for the PTM Requester role is indicated by setting the PTM Requester Capable bit in the PTM Capability register.
- PTM Requesters are permitted to request PTM Master Time only when PTM is enabled. The mechanism for directing a PTM Requester to issue such a request is implementation specific.
 - Upstream Ports obtain PTM Master Time via PTM dialogs as described in section 2.2.8.10.
 - The mechanism by which Root Complex Integrated Endpoints request PTM Master Time is implementation specific.
- Once having issued a PTM Request Message, the Upstream Port must not issue another PTM Request Message prior to the receipt of a PTM Response Message, PTM ResponseD Message, Reset, or the passage of 100 µs without a corresponding PTM Message from the Downstream Port.
- Upon receiving a PTM Response, the Upstream Port must wait at least 1µs before issuing another PTM Request Message.
- For multi-function devices (MFDs) containing multiple PTM Requesters, the Upstream Port associated with that MFD must issue a single PTM dialog during each PTM context refresh. PTM Requesters within the MFD maintain their individual PTM contexts using this one, Device-wide PTM dialog. The mechanism for refreshing multiple PTM contexts from one PTM dialog is implementation specific.
- It is strongly recommended that an Upstream Port invalidate its internal PTM context when the relationship between PTM Master Time and the Upstream Port's local time changes, as determined by implementation specific criteria. For example, this may occur as a result of a transition to a non-D0 state or due to accumulated PPM drift. These events are grouped under the label "Local Time Invalidation Event" in Figure 6-26.

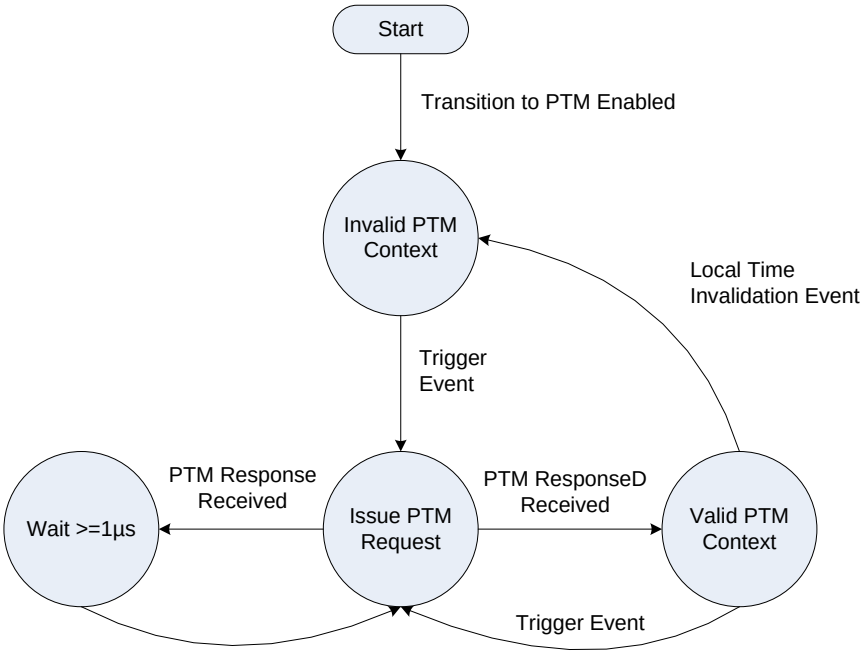


Figure 6-26: PTM Requester Operation

6.22.3.2. PTM Responder Role

- Support for the PTM Responder role is indicated by setting the PTM Responder Capable bit in the PTM Capability register.
- Switches and Root Complexes are permitted to implement the PTM Responder Role.
 - A PTM capable Switch, when enabled for PTM by setting the PTM Enable bit in the PTM Control register associated with the Switch Upstream Port, must respond to all PTM Request Messages received at any of its Downstream Ports.
 - The mechanism by which Root Complexes communicate PTM Master Time to Root Complex Integrated Endpoints is implementation specific.
- PTM Responders must populate PTM ResponseD Messages as follows (refer to Figure 6-24 and the accompanying implementation note):
 - The PTM Master Time field is a 64-bit value containing the value of PTM Master Time at the receipt of the PTM Request Message for the current PTM Dialog. In Figure 6-24, for the 2nd PTM dialog, this is the PTM Master Time at time t2'.
 - The Propagation Delay field is a 32-bit value containing the interval between the receipt of the PTM Request Message and the transmission of the PTM Response Message for the previous PTM dialog. In Figure 6-24, for the 2nd PTM dialog, this is the time interval between t2 and t3 captured during the 1st PTM dialog.
 - The unit of measurement for both fields is one ns.
- Switch Downstream Ports and Root Ports acting as PTM Responders must respond to each PTM Request Message received at their Downstream Ports with either PTM Response or PTM ResponseD according to the following rules:
 - A PTM Responder must not send a PTM Response or PTM ResponseD Message without first receiving a PTM Request Message.
 - Upon receipt of a PTM Request Message, a PTM Responder must attempt to issue a PTM Response or PTM ResponseD Message within 10 µs.
 - A PTM Responder must issue PTM Response when the Downstream Port does not have historical timestamps (t3 – t2) with which to fulfill a PTM Request Message.
 - A PTM Responder must issue PTM ResponseD when it has stored copies of the values required to populate the PTM ResponseD Message: historical timestamps (t3 –t2) and the PTM Master Time at the receipt of the most recent PTM Request Message (time t2').
 - A PTM Responder is permitted to issue PTM Response when it has stored copies of the historical timestamps (t3 – t2) but must request the PTM Master Time from elsewhere. In this case, it is permitted to issue PTM Response messages in response to PTM Request Messages while it retrieves the PTM Master Time if that retrieval is expected to take more than 10 µs.
 - The perceived granularity of the historical timestamps and PTM Master Time values transmitted by a PTM Responder must not exceed that reported in the Local Clock Granularity field of the PTM Capability register.

6.22.3.3. PTM Time Source Role -- Rules Specific to Switches

In addition to the requirements listed above for the PTM Requester and PTM Responder Roles, Switches must follow these requirements:

- When the Upstream Port is associated with a multi-Function device, only a single Function associated with that Upstream Port is permitted to implement the PTM Extended Capability structure. For Switches, all PTM functionality associated with the Switch must be controlled through that structure. It is not required that the Function implementing the PTM Extended Capability structure be the Switch Upstream Port Function.
- The PTM Extended Capability structure for a Switch must indicate support for both the PTM Requester and PTM Responder roles.
- The PTM Extended Capability in the Upstream Port controls all Switches in that Upstream Port.
- A Switch is permitted to act as a PTM Root, or to issue PTM Requests on its Upstream Port to obtain the PTM Master Time for use in fulfilling PTM Requests received at its Downstream Ports. In the latter case the Switch must account for any internal delays within the Switch.
- A Switch is permitted to maintain a local PTM context for use in fulfilling PTM Requests received on its Downstream Ports.
- A Switch which is not acting as a PTM Root must invalidate its local context no more than 10 ms from the last PTM dialog on its Upstream Port. The Switch must then refresh its local PTM context prior to issuing further PTM ResponseD Messages on its Downstream Ports. This requirement for periodic refreshes is optional if it is guaranteed by implementation-specific means that the Switch's local clock is phase locked with PTM Master Time.
- Any Switch implementing a local clock for the purpose of maintaining a local PTM context must report the granularity of this clock as defined in the PTM Capabilities structure (section 7.32.3).

Formatted: Font color: Black

IMPLEMENTATION NOTE

PTM Timestamp Capture Mechanisms

PTM uses services from both the Data Link and Transaction Layers. Accuracy requires that time measurements be taken as close to the Physical Layer as possible. Conversely, the messaging protocol itself properly belongs to the Transaction Layer. The PTM message protocol applies to a single Link, where the Upstream Port is the requestor and the Downstream Port is the responder.

Formatted: Font color: Red

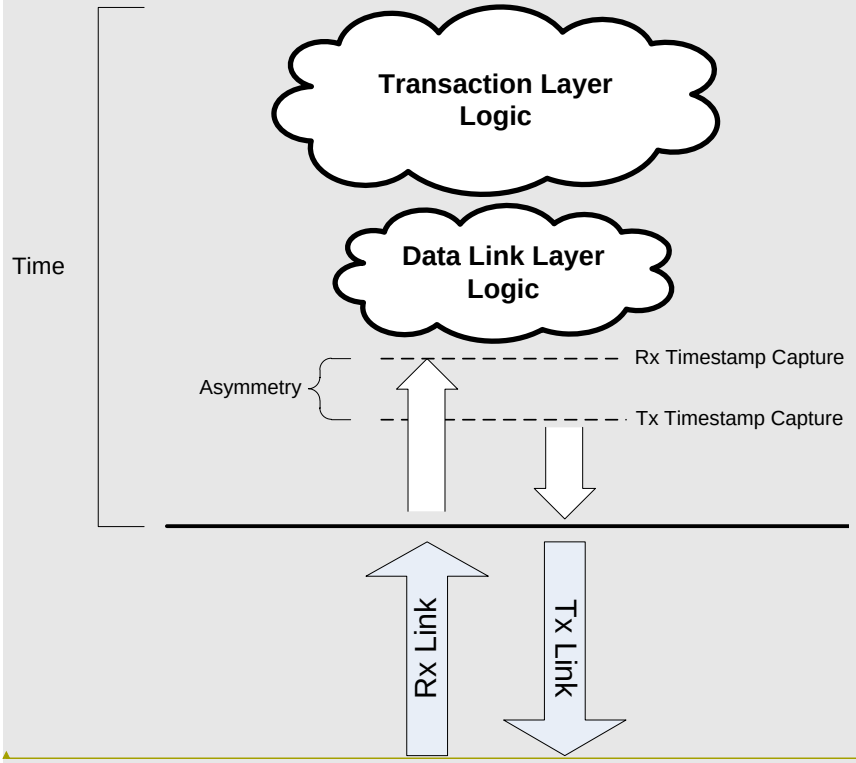


Figure 6-27: PTM Timestamp Capture Example

Figure 6-27 illustrates how to select suitable timestamp capture points. For some implementations, the logic within the Transaction Layer and Data Link Layers is non-deterministic. Implementation details and current conditions have considerable impact on exactly when a particular packet may encounter any particular processing step. This makes it effectively impossible to capture any timestamp that accurately records the time of a particular physical event if timestamps are captured in the higher layers.

6.23. Readiness Notifications (RN)

Readiness Notifications (RN) is intended to reduce the time software needs to wait before issuing Configuration Requests to a Device or Function following DRS Events or FRS Events. RN includes both the Device Readiness Status (DRS) and Function Readiness Status (FRS) mechanisms. These mechanisms provide a direct indication of Configuration-Readiness (see Terms and Acronyms entry for "Configuration-Ready"). When used, DRS and FRS allow an improved behaviour over the CRS mechanism, and eliminate its associated periodic polling time of up to 1 second following a reset.

It is permitted that system software/firmware provide mechanisms that supersede the FRS and/or DRS mechanisms, however such software/firmware mechanisms are outside the scope of this specification.

IMPLEMENTATION NOTE

Optimizing Configuration Readiness

It is strongly recommended that implementers of system firmware/software avoid unnecessary delays wherever possible. It is strongly recommended that hardware be designed to eliminate or minimize required delays, and to make full use of the mechanisms provided in this specification and related specifications to communicate what, if any, delays are required. Hardware implementers should appropriately document implementation behavior to enable system firmware/software to implement optimal behaviors.

Even with good documentation, some cases may at first appear problematic – for example, how can system firmware benefit from the Device Readiness Status (DRS) mechanism, when it is necessary to read from Root Port Configuration space to do so? In such cases, platform specific knowledge is required, i.e. that the Root Port supports Immediate Readiness.

6.23.1. Device Readiness Status (DRS)

When implemented, DRS must be used to indicate when a Device is Configuration-Ready following any of the following Device-level occurrences, which are subsequently referred to as "DRS Events":

- Exit from Cold Reset
- Exit from Warm Reset, Hot Reset, Loopback, or Disabled
- Exit from L2/L3 Ready
- Any other scenario where the Port transitions from DL_Down to DL_Up status.

The DRS Message protocol requirements include the following:

- There is no enable or disable mechanism for DRS. For Downstream Ports that support DRS, the DRS Supported bit in the Link Capabilities 2 register must be Set. For Upstream Ports that support DRS, it is strongly recommended that the DRS Supported bit in the Link Capabilities 2 register be Set. It is expressly permitted for Upstream Ports to send DRS Messages even when the DRS Supported bit is Clear.

Comment [VS227]: Entire section added per D-ECN-Readiness Notifications

Comment [VS228]: Para updated per RC_ECN_RN_27_Aug_2013

Comment [VS229]: Para updated per RC_ECN_RN_27_Aug_2013

Formatted: Font color: Black

Comment [VS230]: RC_ECN_RN_27_Aug_2013

Comment [VS231]: Updated per RC_ECN_RN_27_Aug_2013

- A DRS Message must be transmitted by a DRS-capable Upstream Port following every DL_Down to DL_Up transition when all non-VF Functions on the Logical Bus associated with that Upstream Port become ready.
 - A Type 0 Function is ready when it is Configuration-Ready.
 - A Type 1 Function that is a Switch Upstream Port is ready when it is Configuration-Ready and all Functions on its secondary bus are Configuration-Ready.
 - A Type 1 Function that is not a Switch Upstream Port is ready when the Function itself is Configuration-Ready.
- After a Device transmits a DRS Message, non-VF Functions indicated as Configuration-Ready by that DRS Message must not return Completions with CRS unless a subsequent DRS Event occurs.

Comment [VS232]: Updated per RC_ECN_RN_27_Aug_2013

Additional requirements relating to Switches implementing DRS include:

- Must support DRS functionality in all Ports
- Implementation at each Downstream Port of the DRS Signaling Control field.
- For any physically-integrated Device that appears beneath a Switch Downstream Port, the DRS sent by the Switch does not indicate Configuration Readiness for that Device
 - For such a Device, implementation and use of DRS is independent of the Switch

Additional requirements for Root Ports and Switch Downstream Ports include:

- Implementation of the DRS Message Received bit, which indicates receipt of a DRS Message

6.23.2. Function Readiness Status (FRS)

When implemented, FRS must be used to indicate a specific Function as being Configuration-Ready following any of the following Function-level occurrences, which are subsequently referred to as "FRS Events":

- Function Level Reset (FLR)
- Completion of $D3_{hot}$ to D0 transition
- Setting or Clearing of VF Enable in a PF (SR-IOV)

The FRS Message protocol requirements include the following:

- The Requester ID of the FRS Message must indicate the Function that has changed readiness status (see section 2.2.8.6.4)
- The FRS Reason field in the FRS Message must indicate why that Function changed readiness status.
- After a Function transmits an FRS Message, the indicated Function(s) must not return Completions with CRS unless a subsequent DRS Event or FRS Event occurs

Additional requirements for Switches implementing FRS include:

- Must support FRS functionality in the Upstream Port and all Downstream Ports
- The ability to transmit FRS Messages Upstream when required by the FRS protocol

Additional requirements for Physical Functions (PFs) include:

- The ability to transmit FRS Message Upstream when the VF Enable or VF Disable process completes

Additional requirements for Root Ports and Root Complex Event Collectors implementing FRS include:

- Must implement the FRS Queuing Extended Capability (see Section 7.34)

6.23.3. FRS Queuing

Comment [VS233]: Section updated per RC_ECN_RN_27_Aug_2013

Root Ports and Root Complex Event Collectors that support FRS must implement the FRS Queuing Extended Capability (see Section 7.34).

For a Root Port, the FRS Message Queue contains FRS Messages received by the Root Port or generated by the Root Port.

For a Root Complex Event Collector, the FRS Message Queue contains FRS Messages generated by Root Complex Integrated Endpoints associated with the Root Complex Event Collector (see Section 7.17) or generated by the Root Complex Event Collector.

The FRS Message Queue must satisfy the following requirements:

- The FRS Message Queue must be empty following Reset.
- For a Root Port, the FRS Message Queue must be emptied when the Link goes to DL_Down.
- FRS Messages must be queued in the order received.
- If the FRS Message Queue is not full at the time an FRS Message is received or is internally generated, that FRS Message must be entered in the queue and the FRS Message Received bit must set to 1b.
- If the FRS Message Queue is full at the time an FRS Message is received or is internally generated, that FRS Message must be discarded and the FRS Message Overflow bit must be set to 1b. The pre-existing FRS Message Queue must be preserved.
- The oldest FRS Message must be visible in the FRS Message Queue register (see Section 7.34.4).
- Writing the FRS Message Queue register must remove the oldest element from the queue.
- When either FRS Message Received or FRS Message Overflow transitions from 0b to 1b, an interrupt must be generated if enabled.

7

7. Software Initialization and Configuration

The PCI Express Configuration model supports two Configuration Space access mechanisms:

- PCI compatible Configuration Access Mechanism (CAM)
- PCI Express Enhanced Configuration Access Mechanism (ECAM)

The PCI compatible mechanism supports 100% binary compatibility with PCI 3.0 or later aware operating systems and their corresponding bus enumeration and configuration software.

The enhanced mechanism is provided to increase the size of available Configuration Space and to optimize access mechanisms.

7.1. Configuration Topology

To maintain compatibility with PCI software configuration mechanisms, all PCI Express elements have a PCI-compatible Configuration Space. Each PCI Express Link originates from a logical PCI-PCI Bridge and is mapped into Configuration Space as the secondary bus of this Bridge. The Root Port is a PCI-PCI Bridge structure that originates a PCI Express Link from a PCI Express Root Complex (see Figure 7-1).

A PCI Express Switch is represented by multiple PCI-PCI Bridge structures connecting PCI Express Links to an internal logical PCI bus (see Figure 7-2). The Switch Upstream Port is a PCI-PCI Bridge; the secondary bus of this Bridge represents the Switch's internal routing logic. Switch Downstream Ports are PCI-PCI Bridges bridging from the internal bus to buses representing the Downstream PCI Express Links from a PCI Express Switch. Only the PCI-PCI Bridges representing the Switch Downstream Ports may appear on the internal bus. Endpoints, represented by Type 0 Configuration Space headers, are not permitted to appear on the internal bus.

A PCI Express Endpoint is mapped into Configuration Space as a single Function in a Device, which might contain multiple Functions or just that Function. PCI Express Endpoints and Legacy Endpoints are required to appear within one of the Hierarchy Domains originated by the Root Complex, meaning that they appear in Configuration Space in a tree that has a Root Port as its head. Root Complex Integrated Endpoints and Root Complex Event Collectors do not appear within one of the Hierarchy Domains originated by the Root Complex. These appear in Configuration Space as peers of the Root Ports.

Unless otherwise specified, requirements in the Configuration Space definition for a device apply to single Function devices as well as to each Function individually of a multi-Function device.

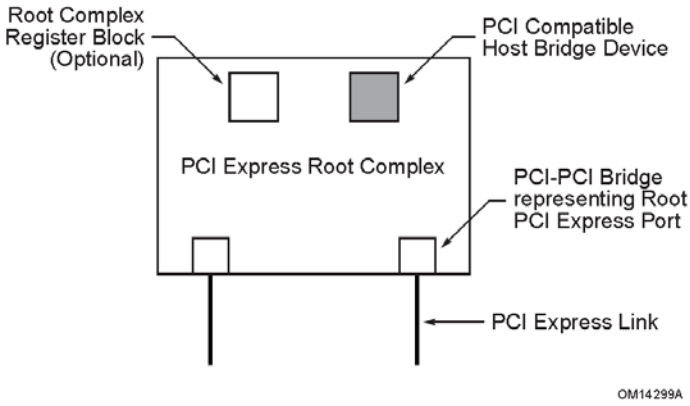


Figure 7-1: PCI Express Root Complex Device Mapping

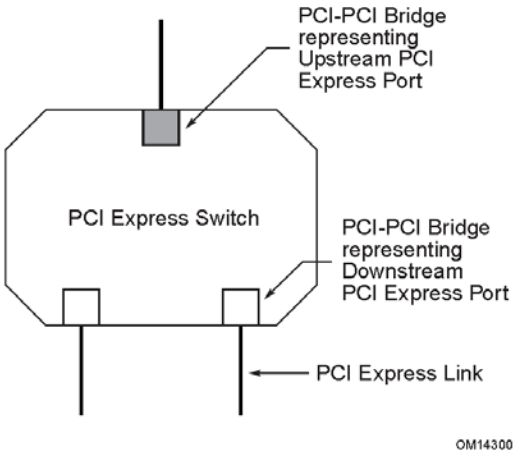


Figure 7-2: PCI Express Switch Device Mapping [104]

7.2. PCI Express Configuration Mechanisms

PCI Express extends the Configuration Space to 4096 bytes per Function as compared to 256 bytes allowed by *PCI Local Bus Specification*. PCI Express Configuration Space is divided into a PCI 3.0 compatible region, which consists of the first 256 bytes of a Function's Configuration Space, and a PCI Express Extended Configuration Space which consists of the remaining Configuration Space (see Figure 7-3). The PCI 3.0 compatible Configuration Space can be accessed using either the mechanism defined in the *PCI Local Bus Specification* or the PCI Express Enhanced Configuration Access Mechanism (ECAM) described later in this section. Accesses made using either access

[104] Future PCI Express Switches may be implemented as a single Switch device component (without the PCI-PCI Bridges) that is not limited by legacy compatibility requirements imposed by existing PCI software.

mechanism are equivalent. The PCI Express Extended Configuration Space can only be accessed by using the ECAM.[105]

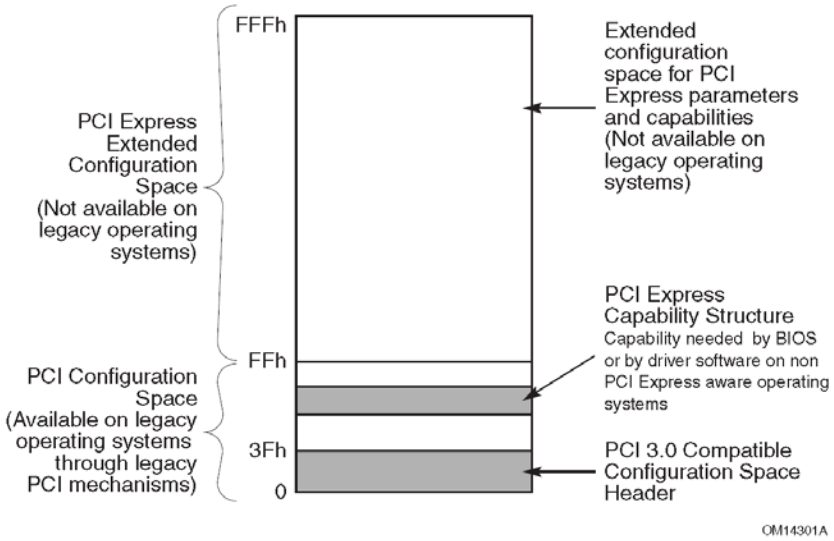


Figure 7-3: PCI Express Configuration Space Layout

7.2.1. PCI 3.0 Compatible Configuration Mechanism

The PCI 3.0 compatible PCI Express Configuration Mechanism supports the PCI Configuration Space programming model defined in the *PCI Local Bus Specification*. By adhering to this model, systems incorporating PCI Express interfaces remain compliant with conventional PCI bus enumeration and configuration software.

In the same manner as PCI 3.0 device Functions, PCI Express device Functions are required to provide a Configuration Space for software-driven initialization and configuration. Except for the differences described in this chapter, the PCI Express Configuration Space headers are organized to correspond with the format and behavior defined in the *PCI Local Bus Specification* (Section 6.1).

The PCI 3.0 compatible Configuration Access Mechanism uses the same Request format as the ECAM. For PCI compatible Configuration Requests, the Extended Register Address field must be all zeros.

[105] The ECAM operates independently from the mechanism defined in the *PCI Local Bus Specification* for generation of configuration transactions; there is no implied ordering between the two.

7.2.2. PCI Express Enhanced Configuration Access Mechanism (ECAM)

For systems that are PC-compatible, or that do not implement a processor-architecture-specific firmware interface standard that allows access to the Configuration Space, the ECAM is required as defined in this section.

For systems that implement a processor-architecture-specific firmware interface standard that allows access to the Configuration Space, the operating system uses the standard firmware interface, and the hardware access mechanism defined in this section is not required. For example, for systems that are compliant with *Developer's Interface Guide for 64-bit Intel Architecture-based Servers (DIG64), Version 2.1*,[106] the operating system uses the SAL firmware service to access the Configuration Space.

In all systems, device drivers are encouraged to use the application programming interface (API) provided by the operating system to access the Configuration Space of its device and not directly use the hardware mechanism.

The ECAM utilizes a flat memory-mapped address space to access device configuration registers. In this case, the memory address determines the configuration register accessed and the memory data updates (for a write) or returns the contents of (for a read) the addressed register. The mapping from memory address space to PCI Express Configuration Space address is defined in Table 7-1.

The size and base address for the range of memory addresses mapped to the Configuration Space are determined by the design of the host bridge and the firmware. They are reported by the firmware to the operating system in an implementation-specific manner. The size of the range is determined by the number of bits that the host bridge maps to the Bus Number field in the configuration address. In Table 7-1, this number of bits is represented as n, where $1 \leq n \leq 8$. A host bridge that maps n memory address bits to the Bus Number field supports Bus Numbers from 0 to $2^n - 1$, inclusive, and the base address of the range is aligned to a $2^{(n+20)}$-byte memory address boundary. Any bits in the Bus Number field that are not mapped from memory address bits must be Clear.

For example, if a system maps three memory address bits to the Bus Number field, the following are all true:

- $n = 3$.
- Address bits A[63:23] are used for the base address, which is aligned to a 2^{23}-byte (8-MB) boundary.
- Address bits A[22:20] are mapped to bits [2:0] in the Bus Number field.
- Bits [7:3] in the Bus Number field are set to Clear.
- The system is capable of addressing Bus Numbers between 0 and 7, inclusive.

A minimum of one memory address bit ($n = 1$) must be mapped to the Bus Number field. Systems are encouraged to map additional memory address bits to the Bus Number field as needed to

[106] *Developer's Interface Guide for 64-bit Intel Architecture-based Servers (DIG64), Version 2.1*, January 2002, www.dig64.org

support a larger number of buses. Systems that support more than 4 GB of memory addresses are encouraged to map eight bits of memory address (n = 8) to the Bus Number field. Note that in systems that include multiple host bridges with different ranges of Bus Numbers assigned to each host bridge, the highest Bus Number for the system is limited by the number of bits mapped by the host bridge to which the highest bus number is assigned. In such a system, the highest Bus Number assigned to a particular host bridge would be greater, in most cases, than the number of buses assigned to that host bridge. In other words, for each host bridge, the number of bits mapped to the Bus Number field, n, must be large enough that the highest Bus Number assigned to each particular bridge must be less than or equal to $2^n - 1$ for that bridge.

In some processor architectures, it is possible to generate memory accesses that cannot be expressed in a single Configuration Request, for example due to crossing a DW aligned boundary, or because a locked access is used. A Root Complex implementation is not required to support the translation to Configuration Requests of such accesses.

Table 7-1: Enhanced Configuration Address Mapping

Memory Address[107]	PCI Express Configuration Space
A[(20 + n – 1):20]	Bus Number 1 ≤ n ≤ 8
A[19:15]	Device Number
A[14:12]	Function Number
A[11:8]	Extended Register Number
A[7:2]	Register Number
A[1:0]	Along with size of the access, used to generate Byte Enables

Note: for Requests targeting Extended Functions in an ARI Device, A[19:12] represents the (8-bit) Function Number, which replaces the (5-bit) Device Number and (3-bit) Function Number fields above.

The system hardware must provide a method for the system software to guarantee that a write transaction using the ECAM is completed by the completer before system software execution continues.

[107] This address refers to the byte-level address from a software point of view.



IMPLEMENTATION NOTE

Ordering Considerations for the Enhanced Configuration Access Mechanism

The ECAM converts memory transactions from the host CPU into Configuration Requests on the PCI Express fabric. This conversion potentially creates ordering problems for the software, because writes to memory addresses are typically posted transactions but writes to Configuration Space are not posted on the PCI Express fabric.

Generally, software does not know when a posted transaction is completed by the completer. In those cases in which the software must know that a posted transaction is completed by the completer, one technique commonly used by the software is to read the location that was just written. For systems that follow the PCI ordering rules throughout, the read transaction will not complete until the posted write is complete. However, since the PCI ordering rules allow non-posted write and read transactions to be reordered with respect to each other, the CPU must wait for a non-posted write to complete on the PCI Express fabric to be guaranteed that the transaction is completed by the completer.

As an example, software may wish to configure a device Function's Base Address register by writing to the device using the ECAM, and then read a location in the memory-mapped range described by this Base Address register. If the software were to issue the memory-mapped read before the ECAM write was completed, it would be possible for the memory-mapped read to be re-ordered and arrive at the device before the Configuration Write Request, thus causing unpredictable results.

To avoid this problem, processor and host bridge implementations must ensure that a method exists for the software to determine when the write using the ECAM is completed by the completer.

This method may simply be that the processor itself recognizes a memory range dedicated for mapping ECAM accesses as unique, and treats accesses to this range in the same manner that it would treat other accesses that generate non-posted writes on the PCI Express fabric, i.e., that the transaction is not posted from the processor's viewpoint. An alternative mechanism is for the host bridge (rather than the processor) to recognize the memory-mapped Configuration Space accesses and not to indicate to the processor that this write has been accepted until the non-posted Configuration Transaction has completed on the PCI Express fabric. A third alternative would be for the processor and host bridge to post the memory-mapped write to the ECAM and for the host bridge to provide a separate register that the software can read to determine when the Configuration Write Request has completed on the PCI Express fabric. Other alternatives are also possible.

IMPLEMENTATION NOTE

Generating Configuration Requests

Because Root Complex implementations are not required to support the generation of Configuration Requests from accesses that cross DW boundaries, or that use locked semantics, software should take care not to cause the generation of such accesses when using the memory-mapped ECAM unless it is known that the Root Complex implementation being used will support the translation.

7.2.2.1. Host Bridge Requirements

For those systems that implement the ECAM, the PCI Express Host Bridge is required to translate the memory-mapped PCI Express Configuration Space accesses from the host processor to PCI Express configuration transactions. The use of Host Bridge PCI class code is Reserved for backwards compatibility; host Bridge Configuration Space is opaque to standard PCI Express software and may be implemented in an implementation specific manner that is compatible with PCI Host Bridge Type 0 Configuration Space. A PCI Express Host Bridge is not required to signal errors through a Root Complex Event Collector. This support is optional for PCI Express Host Bridges.

7.2.2.2. PCI Express Device Requirements

Devices must support an additional 4 bits for decoding configuration register access, i.e., they must decode the Extended Register Address[3:0] field of the Configuration Request header.



IMPLEMENTATION NOTE

Device-Specific Registers in Configuration Space

It is strongly recommended that PCI Express devices place no registers in Configuration Space other than those in headers or Capability structures architected by applicable PCI specifications.

Device-specific registers that have legitimate reasons to be placed in Configuration Space (e.g., they need to be accessible before Memory Space is allocated) should be placed in a Vendor-Specific Capability structure (in PCI Compatible Configuration Space) or a Vendor-Specific Extended Capability structure (in PCI Express Extended Configuration Space).

Device-specific registers accessed in the run-time environment by drivers should be placed in Memory Space that is allocated by one or more Base Address registers. Even though PCI Compatible or PCI Express Extended Configuration Space may have adequate room for run-time device-specific registers, placing them there is highly discouraged for the following reasons:

- Not all Operating Systems permit drivers to access Configuration Space directly.
- Some platforms provide access to Configuration Space only via firmware calls, which typically have substantially lower performance compared to mechanisms for accessing Memory Space.
- Even on platforms that provide direct access to a memory-mapped PCI Express Enhanced Configuration Mechanism, performance for accessing Configuration Space will typically be significantly lower than for accessing Memory Space since:

Configuration Reads and Writes must usually be DWORD or smaller in size,

Configuration Writes are usually not posted by the platform, and

Some platforms support only one outstanding Configuration Write at a time.

7.2.3. Root Complex Register Block

A Root Port or Root Complex Integrated Endpoint may be associated with an optional 4096-byte block of memory mapped registers referred to as the Root Complex Register Block (RCRB). These registers are used in a manner similar to Configuration Space and can include PCI Express Extended Capabilities and other implementation specific registers that apply to the Root Complex. The structure of the RCRB is described in Section 7.9.2.

Multiple Root Ports or internal devices are permitted to be associated with the same RCRB. The RCRB memory-mapped registers must not reside in the same address space as the memory-mapped Configuration Space or Memory Space.

A Root Complex implementation is not required to support accesses to an RCRB that cross DWORD aligned boundaries or accesses that use locked semantics.



IMPLEMENTATION NOTE

Accessing Root Complex Register Block

Because Root Complex implementations are not required to support accesses to a RCRB that cross DW boundaries, or that use locked semantics, software should take care not to cause the generation of such accesses when accessing a RCRB unless the Root Complex will support the access.

7.3. Configuration Transaction Rules

7.3.1. Device Number

With non-ARI Devices, PCI Express components are restricted to implementing a single Device Number on their primary interface (Upstream Port), but are permitted to implement up to eight independent Functions within that Device Number. Each internal Function is selected based on decoded address information that is provided as part of the address portion of Configuration Request packets.

Downstream Ports that do not have ARI Forwarding enabled must associate only Device 0 with the device attached to the Logical Bus representing the Link from the Port. Configuration Requests targeting the Bus Number associated with a Link specifying Device Number 0 are delivered to the device attached to the Link; Configuration Requests specifying all other Device Numbers (1-31) must be terminated by the Switch Downstream Port or the Root Port with an Unsupported Request Completion Status (equivalent to Master Abort in PCI). Non-ARI Devices must not assume that Device Number 0 is associated with their Upstream Port, but must capture their assigned Device Number as discussed in Section 2.2.6.2. Non-ARI Devices must respond to all Type 0 Configuration Read Requests, regardless of the Device Number specified in the Request.

Switches, and components wishing to incorporate more than eight Functions at their Upstream Port, are permitted to implement one or more "virtual switches" represented by multiple Type 1 (PCI-PCI Bridge) Configuration Space headers as illustrated in Figure 7-2. These virtual switches serve to allow fan-out beyond eight Functions. Since Switch Downstream Ports are permitted to appear on any Device Number, in this case all address information fields (Bus, Device, and Function Numbers) must be completely decoded to access the correct register. Any Configuration Request targeting an unimplemented Bus, Device, or Function must return a Completion with Unsupported Request Completion Status.

With an ARI Device, its Device Number is implied to be 0 rather than specified by a field within an ID. The traditional 5-bit Device Number and 3-bit Function Number fields in its associated Routing IDs, Requester IDs, and Completer IDs are interpreted as a single 8-bit Function Number. See Section 6.13. Any Type 0 Configuration Request targeting an unimplemented Function in an ARI Device must be handled as an Unsupported Request.

If an ARI Downstream Port has ARI Forwarding enabled, the logic that determines when to turn a Type 1 Configuration Request into a Type 0 Configuration Request no longer enforces a restriction on the traditional Device Number field being 0.

The following section provides details of the Configuration Space addressing mechanism.

7.3.2. Configuration Transaction Addressing

PCI Express Configuration Requests use the following addressing fields:

- Bus Number – PCI Express maps logical PCI Bus Numbers onto PCI Express Links such that PCI 3.0 compatible configuration software views the Configuration Space of a PCI Express Hierarchy as a PCI hierarchy including multiple bus segments.
- Device Number – Device Number association is discussed in Section 7.3.1. When an ARI Device is targeted and the Downstream Port immediately above it is enabled for ARI Forwarding, the Device Number is implied to be 0, and the traditional Device Number field is used instead as part of an 8-bit Function Number field. See Section 6.13.
- Function Number – PCI Express also supports multi-Function devices using the same discovery mechanism as PCI 3.0. A multi-Function Device must fully decode the Function Number field. It is strongly recommended that a single-Function Device also fully decode the Function Number field. With ARI Devices, discovery and enumeration of Extended Functions require ARI-aware software. See Section 6.13.
- Extended Register Number and Register Number – Specify the Configuration Space address of the register being accessed (concatenated such that the Extended Register Number forms the more significant bits).

Comment [VS234]: Errata A18

7.3.3. Configuration Request Routing Rules

For Endpoints, the following rules apply:

- If Configuration Request Type is 1,
 - Follow the rules for handling Unsupported Requests
- If Configuration Request Type is 0,
 - Determine if the Request addresses a valid local Configuration Space of an implemented Function
 - If so, process the Request
 - If not, follow rules for handling Unsupported Requests

For Root Ports, Switches, and PCI Express-PCI Bridges, the following rules apply:

- Propagation of Configuration Requests from Downstream to Upstream as well as peer-to-peer are not supported
 - Configuration Requests are initiated only by the Host Bridge
- If Configuration Request Type is 0,
 - Determine if the Request addresses a valid local Configuration Space of an implemented Function
 - If so, process the Request
 - If not, follow rules for handling Unsupported Requests

- If Configuration Request Type is 1, apply the following tests, in sequence, to the Bus Number and Device Number fields:
 - If in the case of a PCI Express-PCI Bridge, equal to the Bus Number assigned to secondary PCI bus or, in the case of a Switch or Root Complex, equal to the Bus Number and decoded Device Numbers assigned to one of the Root (Root Complex) or Downstream Ports (Switch),
 - Transform the Request to Type 0 by changing the value in the Type[4:0] field of the Request (see Table 2-3) – all other fields of the Request remain unchanged
 - Forward the Request to that Downstream Port (or PCI bus, in the case of a PCI Express-PCI Bridge)
 - If not equal to the Bus Number of any of Downstream Ports or secondary PCI bus, but in the range of Bus Numbers assigned to either a Downstream Port or a secondary PCI bus,
 - Forward the Request to that Downstream Port interface without modification
 - Else (none of the above)
 - The Request is invalid – follow the rules for handling Unsupported Requests
- PCI Express-PCI Bridges must terminate as Unsupported Requests any Configuration Requests for which the Extended Register Address field is non-zero that are directed towards a PCI bus that does not support Extended Configuration Space.

 Note: This type of access is a consequence of a programming error.

Additional rule specific to Root Complexes:

- Configuration Requests addressing Bus Numbers assigned to devices within the Root Complex are processed by the Root Complex
 - The assignment of Bus Numbers to the devices within a Root Complex may be done in an implementation specific way.

For all types of devices:

All other Configuration Space addressing fields are decoded according to the *PCI Local Bus Specification*.

7.3.4. PCI Special Cycles

PCI Special Cycles (see the *PCI Local Bus Specification* for details) are not directly supported by PCI Express. PCI Special Cycles may be directed to PCI bus segments behind PCI Express-PCI Bridges using Type 1 Configuration Cycles as described in *PCI Local Bus Specification*.

7.4. Configuration Register Types

Configuration register fields are assigned one of the attributes described in Table 7-2. All PCI Express components, with the exception of the Root Complex and system-integrated devices, initialize register fields to specified default values. Root Complexes and system-integrated devices initialize register fields as required by the firmware for a particular system implementation.

Table 7-2: Register and Register Bit-Field Types

Register Attribute	Description
HwInit	**Hardware Initialized** – Register bits are initialized by firmware or hardware mechanisms such as pin strapping or serial EEPROM. (System firmware hardware initialization is only allowed for system-integrated devices.) Bits are read-only after initialization and can only be reset (for write-once by firmware) with Fundamental Reset (see Section 6.6.1). HwInit register bits are not modified by an FLR.
RO	**Read-only** – Register bits are read-only and cannot be altered by software. Register bits are permitted to be initialized by hardware mechanisms such as pin strapping or serial EEPROM.
RW	**Read-Write** – Register bits are read-write and are permitted to be either Set or Cleared by software to the desired state.
RW1C	**Write-1-to-clear status** – Register bits indicate status when read. A Set bit indicates a status event which is Cleared by writing a 1b. Writing a 0b to RW1C bits has no effect.
ROS	**Sticky - Read-only** – Register bits are read-only and cannot be altered by software. Bits are neither initialized nor modified by hot reset or FLR. Where noted, devices that consume Aux power must preserve sticky register values when Aux power consumption (via either Aux Power PM Enable or PME_En) is enabled. In these cases, registers are neither initialized nor modified by hot, warm, or cold reset (see Section 6.6).
RWS	**Sticky - Read-Write** – Register bits are read-write and are Set or Cleared by software to the desired state. Bits are neither initialized nor modified by hot reset or FLR. Where noted, devices that consume Aux power must preserve sticky register values when Aux power consumption (via either Aux Power PM Enable or PME_En) is enabled. In these cases, registers are neither initialized nor modified by hot, warm, or cold reset (see Section 6.6).

Register Attribute	Description
RW1CS	**Sticky - Write-1-to-clear status** – Registers indicate status when read. A Set bit indicates a status event which is Cleared by writing a 1b. Writing a 0b to RW1CS bits has no effect. Bits are neither initialized nor modified by hot reset or FLR. Where noted, devices that consume Aux power must preserve sticky register values when Aux power consumption (via either Aux Power PM Enable or PME_En) is enabled. In these cases, registers are neither initialized nor modified by hot, warm, or cold reset (see Section 6.6).
RsvdP	**Reserved and Preserved** – Reserved for future RW implementations. Registers are read-only and must return zero when read. Software must preserve the value read for writes to bits.
RsvdZ	**Reserved and Zero** – Reserved for future RW1C implementations. Registers are read-only and must return zero when read. Software must use 0b for writes to bits.

7.5. PCI-Compatible Configuration Registers

The first 256 bytes of the PCI Express Configuration Space form the PCI 3.0 compatibility region. This region completely aliases the PCI 3.0 Configuration Space of the Function. Legacy PCI devices can also be accessed with the ECAM without requiring any modifications to the device hardware or device driver software. This section establishes the mapping between PCI 3.0 and PCI Express for format and behavior of PCI 3.0 compatible registers.

All registers and fields not described in this section have the same definition as in the *PCI Local Bus Specification*. Layout of the Configuration Space and format of individual configuration registers are depicted following the little-endian convention used in the *PCI Local Bus Specification.*

7.5.1. Type 0/1 Common Configuration Space

Figure 7-4 details allocation for common register fields of PCI 3.0 Type 0 and Type 1 Configuration Space headers for PCI Express device Functions.

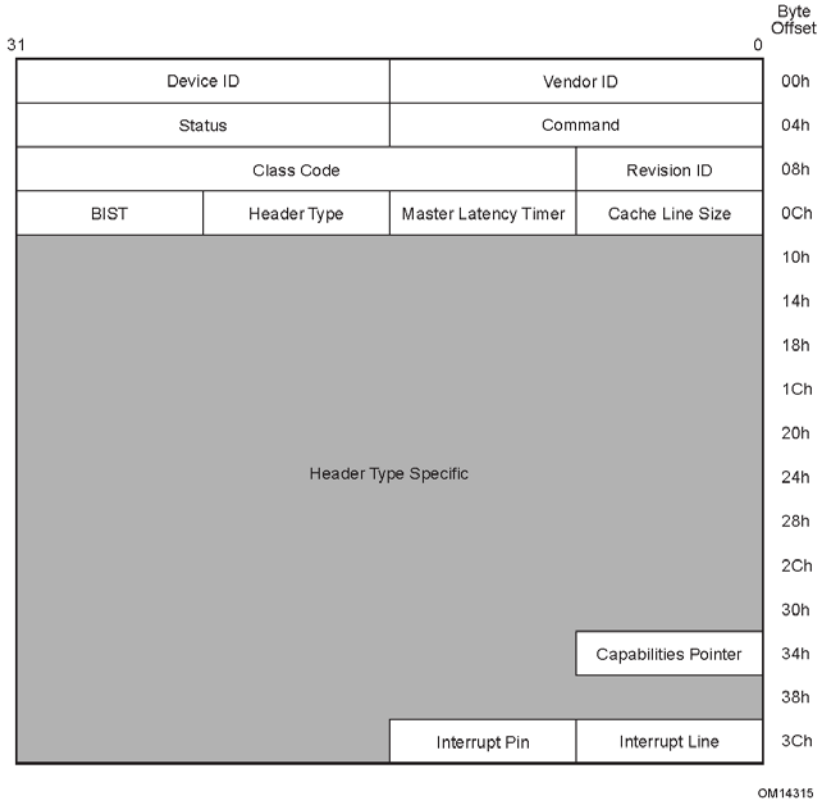


Figure 7-4: Common Configuration Space Header

These registers are defined for both Type 0 and Type 1 Configuration Space headers. The PCI Express-specific interpretation of these registers is defined in this section.

7.5.1.1. Command Register (Offset 04h)

Table 7-3 establishes the mapping between PCI 3.0 and PCI Express for the PCI 3.0 Configuration Space Command register. For PCI Express to PCI/PCI-X Bridges, refer to the *PCI Express to PCI/PCI-X Bridge Specification* for additional requirements for some bits in this register.

Table 7-3: Command Register

Bit Location	Register Description	Attributes
2	**Bus Master Enable** – Controls the ability of a PCI Express Endpoint to issue Memory[108] and I/O Read/Write Requests, and the ability of a Root or Switch Port to forward Memory and I/O Read/Write Requests in the Upstream direction *Endpoints:* When this bit is Set, the PCI Express Function is allowed to issue Memory or I/O Requests. When this bit is Clear, the PCI Express Function is not allowed to issue any Memory or I/O Requests. Note that as MSI/MSI-X interrupt Messages are in-band memory writes, setting the Bus Master Enable bit to 0b disables MSI/MSI-X interrupt Messages as well. Requests other than Memory or I/O Requests are not controlled by this bit. Default value of this bit is 0b. This bit is hardwired to 0b if a Function does not generate Memory or I/O Requests. *Root and Switch Ports:* This bit controls forwarding of Memory or I/O Requests by a Switch or Root Port in the Upstream direction. When this bit is 0b, Memory and I/O Requests received at a Root Port or the Downstream side of a Switch Port must be handled as Unsupported Requests (UR), and for Non-Posted Requests a Completion with UR completion status must be returned. This bit does not affect forwarding of Completions in either the Upstream or Downstream direction. The forwarding of Requests other than Memory or I/O Requests is not controlled by this bit. Default value of this bit is 0b.	RW
3	**Special Cycle Enable** – Does not apply to PCI Express and must be hardwired to 0b.	RO
4	**Memory Write and Invalidate** – Does not apply to PCI Express and must be hardwired to 0b.	RO

[108] The AtomicOp Requester Enable bit in the Device Control 2 register must also be Set in order for an AtomicOp Requester to initiate AtomicOp Requests, which are Memory Requests.

Bit Location	Register Description	Attributes
5	**VGA Palette Snoop** – Does not apply to PCI Express and must be hardwired to 0b.	RO
6	**Parity Error Response** – See Section 7.5.1.8. This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Status register. A Root Complex Integrated Endpoint that is not associated with a Root Complex Event Collector is permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RW
7	**IDSEL Stepping/Wait Cycle Control** – Does not apply to PCI Express and must be hardwired to 0b.	RO
8	**SERR# Enable** – See Section 7.5.1.8. When Set, this bit enables reporting of Non-fatal and Fatal errors detected by the Function to the Root Complex. Note that errors are reported if enabled either through this bit or through the PCI Express specific bits in the Device Control register (see Section 7.8.4). In addition, for Functions with Type 1 Configuration Space headers, this bit controls transmission by the primary interface of ERR_NONFATAL and ERR_FATAL error Messages forwarded from the secondary interface. This bit does not affect the transmission of forwarded ERR_COR messages. A Root Complex Integrated Endpoint that is not associated with a Root Complex Event Collector is permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RW
9	**Fast Back-to-Back Transactions Enable** – Does not apply to PCI Express and must be hardwired to 0b.	RO

Bit Location	Register Description	Attributes
10	**Interrupt Disable** – Controls the ability of a PCI Express Function to generate INTx interrupts. When Set, Functions are prevented from asserting INTx interrupts. Any INTx emulation interrupts already asserted by the Function must be deasserted when this bit is Set. As described in Section 2.2.8.1, INTx interrupts use virtual wires that must, if asserted, be deasserted using the appropriate Deassert_INTx message(s) when this bit is Set. Only the INTx virtual wire interrupt(s) associated with the Function(s) for which this bit is Set are affected. For Endpoints that generate INTx interrupts, this bit is required. For Endpoints that do not generate INTx interrupts this bit is optional. If not implemented, this bit must be hardwired to 0b. For Root Ports, Switch Ports, and Bridges that generate INTx interrupts on their own behalf, this bit is required. This bit has no effect on interrupts that pass through the Port from the secondary side. For Root Ports, Switch Ports, and Bridges that do not generate INTx interrupts on their own behalf this bit is optional. If not implemented, this bit must be hardwired to 0b. Default value of this bit is 0b.	RW

7.5.1.2. Status Register (Offset 06h)

Table 7-4 establishes the mapping between PCI 3.0 and PCI Express for PCI 3.0 Configuration Space Status register. For PCI Express to PCI/PCI-X Bridges, refer to the *PCI Express to PCI/PCI-X Bridge Specification* for additional requirements for some bits and fields in this register.

Table 7-4: Status Register

Bit Location	Register Description	Attributes
3	**Interrupt Status** – When Set, indicates that an INTx emulation interrupt is pending internally in the Function. Note that INTx emulation interrupts forwarded by Root and Switch Ports from devices Downstream of the Root or Switch Port are not reflected in this bit. Default value of this bit is 0b.	RO
4	**Capabilities List** – Indicates the presence of an Extended Capability list item. Since all PCI Express device Functions are required to implement the PCI Express Capability structure, this bit must be hardwired to 1b.	RO
5	**66 MHz Capable** – Does not apply to PCI Express and must be hardwired to 0b.	RO
7	**Fast Back-to-Back Transactions Capable** – Does not apply to PCI Express and must be hardwired to 0b.	RO

Bit Location	Register Description	Attributes
8	**Master Data Parity Error** – See Section 7.5.1.8. This bit is Set by an Endpoint Function if the Parity Error Response bit in the Command register is 1b and either of the following two conditions occurs: • Endpoint receives a Poisoned Completion • Endpoint transmits a Poisoned Request This bit is Set by a Root Port, Switch Upstream Port, or Switch Downstream Port if the Parity Error Response bit in the Command register is 1b and either of the following two conditions occurs: • Port receives a Poisoned Completion going Downstream • Port transmits a Poisoned Request Upstream If the Parity Error Response bit is 0b, this bit is never Set. Default value of this bit is 0b.	RW1C
10:9	**DEVSEL Timing** – Does not apply to PCI Express and must be hardwired to 00b.	RO
11	**Signaled Target Abort** – See Section 7.5.1.8. This bit is Set when a Function completes a Posted or Non-Posted Request as a Completer Abort error. This applies to a Function with a Type 1 Configuration header when the Completer Abort was generated by its Primary Side. Default value of this bit is 0b.	RW1C
12	**Received Target Abort** – See Section 7.5.1.8. This bit is Set when a Requester receives a Completion with Completer Abort Completion Status. On a Function with a Type 1 Configuration header, the bit is Set when the Completer Abort is received by its Primary Side. Default value of this bit is 0b.	RW1C
13	**Received Master Abort** – See Section 7.5.1.8. This bit is Set when a Requester receives a Completion with Unsupported Request Completion Status. On a Function with a Type 1 Configuration header, the bit is Set when the Unsupported Request is received by its Primary Side. Default value of this bit is 0b.	RW1C
14	**Signaled System Error** – See Section 7.5.1.8. This bit is Set when a Function sends an ERR_FATAL or ERR_NONFATAL Message, and the SERR# Enable bit in the Command register is 1. Default value of this bit is 0b.	RW1C

Bit Location	Register Description	Attributes
15	**Detected Parity Error** – See Section 7.5.1.8. This bit is Set by a Function whenever it receives a Poisoned TLP, regardless of the state the Parity Error Response bit in the Command register. On a Function with a Type 1 Configuration header, the bit is Set when the Poisoned TLP is received by its Primary Side. Default value of this bit is 0b.	RW1C

7.5.1.3. Cache Line Size Register (Offset 0Ch)

The Cache Line Size register is set by the system firmware or the operating system to system cache line size. However, note that legacy PCI 3.0 software may not always be able to program this field correctly especially in the case of Hot-Plug devices. This field is implemented by PCI Express devices as a read-write field for legacy compatibility purposes but has no effect on any PCI Express device behavior. For PCI Express to PCI/PCI-X Bridges, refer to the *PCI Express to PCI/PCI-X Bridge Specification* for additional requirements for this register. The default value of this register is 00h.

7.5.1.4. Latency Timer Register (Offset 0Dh)

This register is also referred to as Primary Latency Timer for Type 1 Configuration Space header Functions. The Latency Timer does not apply to PCI Express. This register must be hardwired to 00h.

7.5.1.5. Header Type Register (Offset 0Eh)

Comment [VS235]: Errata A18

This register identifies the layout of the second part of the predefined header (beginning at byte 10h in Configuration Space) and also whether or not the Device might contain multiple Functions.

Table 7-5: Header Type Register

Bit Location	Register Description	Attributes
7	**Multi-Function Device** – When Set, indicates that the Device may contain multiple Functions, but not necessarily. Software is permitted to probe for Functions other than Function 0. When Clear, software must not probe for Functions other than Function 0 unless explicitly indicated by another mechanism, such as an ARI or SR-IOV Capability structure. Except where stated otherwise, it is recommended that this bit be Set if there are multiple Functions, and Clear if there is only one Function.	RO

~~7.5.1.5.~~7.5.1.6. Interrupt Line Register (Offset 3Ch)

As in PCI 3.0, the Interrupt Line register communicates interrupt line routing information. The register is read/write and must be implemented by any Function that uses an interrupt pin (see following description). Values in this register are programmed by system software and are system architecture specific. The Function itself does not use this value; rather the value in this register is used by device drivers and operating systems.

~~7.5.1.6.~~7.5.1.7. Interrupt Pin Register (Offset 3Dh)

The Interrupt Pin register is a read-only register that identifies the legacy interrupt Message(s) the Function uses (see Section 6.1 for further details). Valid values are 01h, 02h, 03h, and 04h that map to legacy interrupt Messages for INTA, INTB, INTC, and INTD respectively. A value of 00h indicates that the Function uses no legacy interrupt Message(s).

~~7.5.1.7.~~7.5.1.8. Error Registers

The Error Control/Status register bits in the Command and Status registers (see Section 7.5.1.1 and Section 7.5.1.2 respectively) and the Bridge Control and Secondary Status registers of Type 1 Configuration Space header Functions (see Section 7.5.3.6 and Section 7.5.3.4 respectively) control PCI-compatible error reporting for both PCI and PCI Express device Functions. Mapping of PCI Express errors onto PCI errors is also discussed in Section 6.2.7.1. In addition to the PCI-compatible error control and status, PCI Express error reporting may be controlled separately from PCI device Functions through the PCI Express Capability structure described in Section 7.8. The PCI-compatible error control and status register fields do not have any effect on PCI Express error reporting enabled through the PCI Express Capability structure. PCI Express device Functions may implement optional advanced error reporting as described in Section 7.10.

For PCI Express Root Ports represented by a PCI 3.0 Type 1 Configuration Space header:

- The primary side Error Control/Status registers apply to errors detected on the internal logic associated with the Root Complex.
- The secondary side Error Control/Status registers apply to errors detected on the Link originating from the Root Port.

For PCI Express Switch Upstream Ports represented by a PCI 3.0 Type 1 Configuration Space header:

- The primary side Error Control/Status registers apply to errors detected on the Upstream Link of the Switch.
- The secondary side Error Control/Status registers apply to errors detected on the internal logic of the Switch.

For PCI Express Switch Downstream Ports represented by a PCI 3.0 Type 1 Configuration Space header:

- The primary side Error Control/Status registers apply to errors detected on the internal logic of the Switch.

- The secondary side Error Control/Status registers apply to errors detected on the Downstream Link originating from the Switch Port.

7.5.2. Type 0 Configuration Space Header

Figure 7-5 details allocation for register fields of PCI 3.0 Type 0 Configuration Space header for PCI Express device Functions.

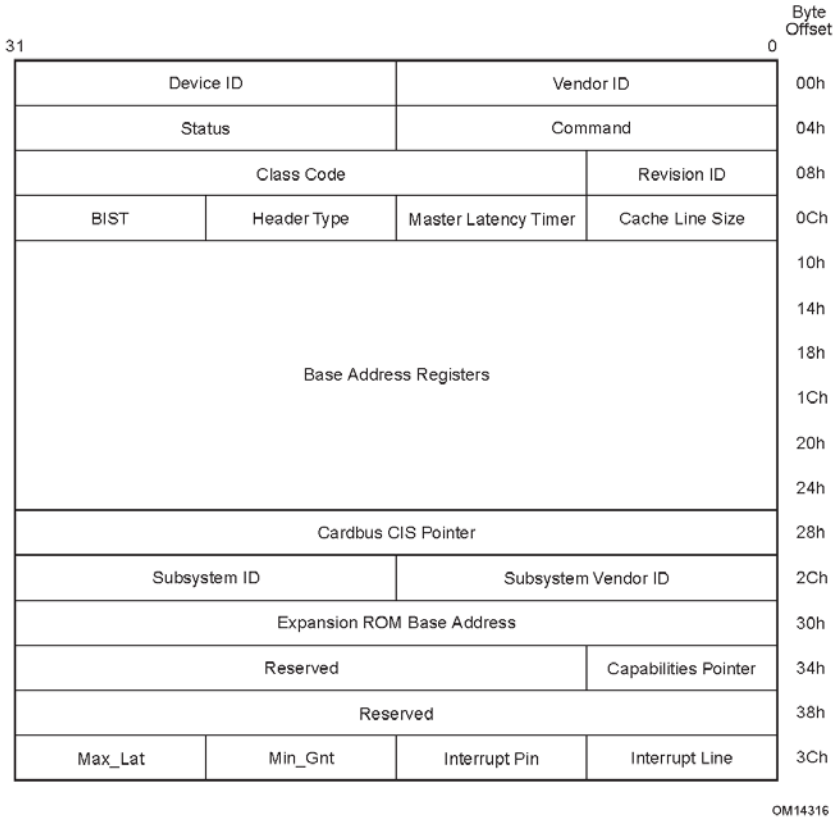


Figure 7-5: Type 0 Configuration Space Header

Section 7.5.1 details the PCI Express-specific registers that are valid for all Configuration Space header types. The PCI Express-specific interpretation of registers specific to Type 0 PCI 3.0 Configuration Space header is defined in this section.

7.5.2.1. Base Address Registers (Offset 10h - 24h)

A PCI Express Function requesting Memory Space through a BAR must set the BAR's Prefetchable bit unless the range contains locations with read side effects or locations in which the Function does

not tolerate write merging. It is strongly encouraged that resources mapped into Memory Space be designed as prefetchable whenever possible. PCI Express Functions other than Legacy Endpoints must support 64-bit addressing for any Base Address register that requests prefetchable Memory Space. The minimum Memory Space address range requested by a BAR is 128 bytes. The attributes for some of the bits in the BAR are affected by the Resizable BAR Capability, if it is implemented.



IMPLEMENTATION NOTE

Additional Guidance on the Prefetchable Bit in Memory Space BARs

PCI Express adapters with Memory Space BARs that request a large amount of non-prefetchable Memory Space (e.g., over 64 MB) may cause shortages of that Space on certain scalable platforms, since many platforms support a total of only 1 GB or less of non-prefetchable Memory Space. This may limit the number of such adapters that can be supported on those platforms. For this reason, it is especially encouraged for BARs requesting large amounts of Memory Space to have their Prefetchable bit Set, since prefetchable Memory Space is more bountiful on most scalable platforms.

While a Memory Space BAR is required have its Prefetchable bit Set if none of the locations within its range have read side effects and all of the locations tolerate write merging, there are system configurations where having the Prefetchable bit Set will still allow correct operation even if those conditions are not met. For those cases, it may make sense for the adapter to have the Prefetchable bit Set in certain candidate BARs so that the system can map those BARs into prefetchable Memory Space in order to avoid non-prefetchable Memory Space shortages.

On PCI Express systems that meet the criteria enumerated below, setting the Prefetchable bit in a candidate BAR will still permit correct operation even if the BAR's range includes some locations that have read side-effects or cannot tolerate write merging. This is primarily due to the fact that PCI Express Memory Reads always contain an explicit length, and PCI Express Switches never prefetch or do byte merging. Generally only 64-bit BARs are good candidates, since only Legacy Endpoints are permitted to set the Prefetchable bit in 32-bit BARs, and most scalable platforms map all 32-bit Memory BARs into non-prefetchable Memory Space regardless of the Prefetchable bit value.

Here are criteria that are sufficient to guarantee correctness for a given candidate BAR:

- ❑ The entire path from the host to the adapter is over PCI Express.
- ❑ No conventional PCI or PCI-X devices do peer-to-peer reads to the range mapped by the BAR.
- ❑ The PCI Express Host Bridge does no byte merging. (This is believed to be true on most platforms.)
- ❑ Any locations with read side-effects are never the target of Memory Reads with the TH bit Set. See Section 2.2.5.
- ❑ The range mapped by the BAR is never the target of a speculative Memory Read, either Host initiated or peer-to-peer.

Comment [VS236]: Errata B2

The above criteria are a simplified set that are sufficient to guarantee correctness. Other less restrictive but more complex criteria may also guarantee correctness, but are outside the scope of this specification.

7.5.2.2. Min_Gnt/Max_Lat Registers (Offset 3Eh/3Fh)

These registers do not apply to PCI Express. They must be read-only and hardwired to 00h.

7.5.3. Type 1 Configuration Space Header

Figure 7-6 details allocation for register fields of PCI 3.0 Type 1 Configuration Space header for Switch and Root Complex virtual PCI Bridges.

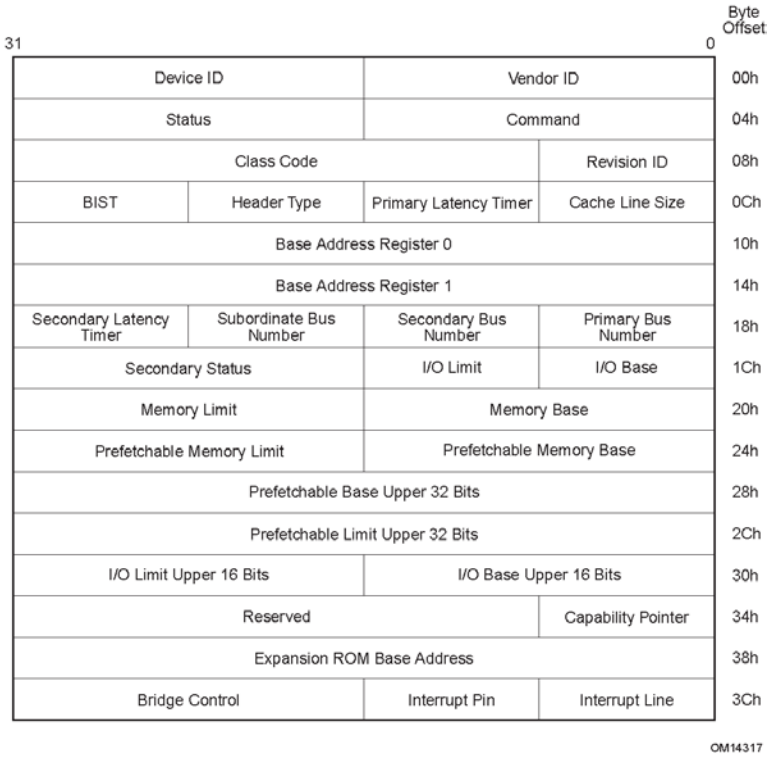


Figure 7-6: Type 1 Configuration Space Header

Section 7.5.1 details the PCI Express-specific registers that are valid for all Configuration Space header types. The PCI Express-specific interpretation of registers specific to Type 1 PCI 3.0 Configuration Space header is defined in this section. Register interpretations described in this section apply to PCI-PCI Bridge structures representing Switch and Root Ports; other device Functions such as PCI Express to PCI/PCI-X Bridges with Type 1 PCI 3.0 Configuration Space headers are not covered by this section.

7.5.3.1. Base Address Registers (Offset 10h/14h)

A PCI Express Function requesting memory resources through a BAR must set the BAR's Prefetchable bit unless the range contains locations with read side effects or locations in which the Function does not tolerate write merging. It is strongly encouraged that memory-mapped resources be designed as prefetchable whenever possible. PCI Express device Functions other than Legacy Endpoints must support 64-bit addressing for any Base Address register that requests prefetchable memory resources. The minimum memory address range requested by a BAR is 128 bytes. The attributes for some of the bits in the BAR are affected by the Resizable BAR Capability, if it is implemented (see Section 7.22).

7.5.3.2. Primary Bus Number (Offset 18h)

Except as noted, this register is not used by PCI Express Functions but must be implemented as read-write for compatibility with legacy software. PCI Express Functions capture the Bus (and Device) Number as described in Section 2.2.6. Refer to *PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0* for exceptions to this requirement.

7.5.3.3. Secondary Latency Timer (Offset 1Bh)

This register does not apply to PCI Express. It must be read-only and hardwired to 00h. For PCI Express to PCI/PCI-X Bridges, refer to the *PCI Express to PCI/PCI-X Bridge Specification* for additional requirements for this register.

7.5.3.4. Secondary Status Register (Offset 1Eh)

Table 7-6 establishes the mapping between PCI 3.0 and PCI Express for PCI 3.0 Configuration Space Secondary Status register. For PCI Express to PCI/PCI-X Bridges, refer to the *PCI Express to PCI/PCI-X Bridge Specification* for additional requirements for some bits and fields in this register.

Table 7-6: Secondary Status Register

Bit Location	Register Description	Attributes
5	**66 MHz Capable** – Does not apply to PCI Express and must be hardwired to 0b.	RO
7	**Fast Back-to-Back Transactions Capable** – Does not apply to PCI Express and must be hardwired to 0b.	RO
8	**Master Data Parity Error** – See Section 7.5.1.8. This bit is Set by a Root Port, Switch Upstream Port, or Switch Downstream Port if the Parity Error Response Enable bit in the Bridge Control register is Set and either of the following two conditions occurs: Port receives a Poisoned Completion coming Upstream Port transmits a Poisoned Request Downstream If the Parity Error Response Enable bit is Clear, this bit is never Set. Default value of this bit is 0b.	RW1C

Bit Location	Register Description	Attributes
10:9	**DEVSEL Timing** – Does not apply to PCI Express and must be hardwired to 00b.	RO
11	**Signaled Target Abort** – See Section 7.5.1.8. This bit is Set when the Secondary Side for Type 1 Configuration Space header Function (for Requests completed by the Type 1 header Function itself) completes a Posted or Non-Posted Request as a Completer Abort error. Default value of this bit is 0b.	RW1C
12	**Received Target Abort** – See Section 7.5.1.8. This bit is Set when the Secondary Side for Type 1 Configuration Space header Function (for Requests initiated by the Type 1 header Function itself) receives a Completion with Completer Abort Completion Status. Default value of this bit is 0b.	RW1C
13	**Received Master Abort** – See Section 7.5.1.8. This bit is Set when the Secondary Side for Type 1 Configuration Space header Function (for Requests initiated by the Type 1 header Function itself) receives a Completion with Unsupported Request Completion Status. Default value of this bit is 0b.	RW1C
14	**Received System Error** – See Section 7.5.1.8. This bit is Set when the Secondary Side for a Type 1 Configuration Space header Function receives an ERR_FATAL or ERR_NONFATAL Message. Default value of this bit is 0b.	RW1C
15	**Detected Parity Error** – See Section 7.5.1.8. This bit is Set by a Function with a Type 1 Configuration header when a Poisoned TLP is received by its Secondary Side, regardless of the state the Parity Error Response Enable bit in the Bridge Control register. Default value of this bit is 0b.	RW1C

7.5.3.5. Prefetchable Memory Base/Limit (Offset 24h)

The Prefetchable Memory Base and Prefetchable Memory Limit registers must indicate that 64-bit addresses are supported, as defined in *PCI-to-PCI Bridge Architecture Specification*.

7.5.3.6. Bridge Control Register (Offset 3Eh)

Table 7-7 establishes the mapping between PCI 3.0 and PCI Express for the PCI 3.0 Configuration Space Bridge Control register. For PCI Express to PCI/PCI-X Bridges, refer to the *PCI Express to PCI/PCI-X Bridge Specification* for additional requirements for some bits in this register.

Table 7-7: Bridge Control Register

Bit Location	Register Description	Attributes
0	**Parity Error Response Enable** – See Section 7.5.1.8. This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Secondary Status register. Default value of this bit is 0b.	RW
1	**SERR# Enable** – See Section 7.5.1.8. This bit controls forwarding of ERR_COR, ERR_NONFATAL and ERR_FATAL from secondary to primary. Default value of this bit is 0b.	RW
5	**Master Abort Mode** – Does not apply to PCI Express and must be hardwired to 0b.	RO
6	**Secondary Bus Reset** – Setting this bit triggers a hot reset on the corresponding PCI Express Port. Software must ensure a minimum reset duration (T_{rst}) as defined in the *PCI Local Bus Specification*. Software and systems must honor first-access-following-reset timing requirements defined in Section 6.6., unless the Readiness Notifications mechanism (see Section 6.23) is used or if the Immediate Readiness bit in the relevant Function's Status Register is Set. Port configuration registers must not be changed, except as required to update Port status. Default value of this bit is 0b.	RW
7	**Fast Back-to-Back Transactions Enable** – Does not apply to PCI Express and must be hardwired to 0b.	RO
8	**Primary Discard Timer** – Does not apply to PCI Express and must be hardwired to 0b.	RO
9	**Secondary Discard Timer** – Does not apply to PCI Express and must be hardwired to 0b.	RO
10	**Discard Timer Status** – Does not apply to PCI Express and must be hardwired to 0b.	RO
11	**Discard Timer SERR# Enable** – Does not apply to PCI Express and must be hardwired to 0b.	RO

Comment [VS237]: D-ECN-Readiness Notifications

7.6. PCI Power Management Capability Structure

This structure is required for all PCI Express device Functions. This Capability is defined in the *PCI Bus Power Management Interface Specification, Revision. 1.2*. The functionality associated with this structure is the same for PCI Express as it is for conventional PCI, and only the added requirements associated with PCI Express are included here.

PCI Express device Functions are required to support D0 and D3 device states (see Section 5.1.1); PCI-PCI Bridge structures representing PCI Express Ports as described in Section 7.1 are required to indicate PME Message passing capability due to the in-band nature of PME messaging for PCI Express.

The PME_Status bit for the PCI-PCI Bridge structure representing PCI Express Ports, however, is only Set when the PCI-PCI Bridge Function is itself generating a PME. The PME_Status bit is not Set when the Bridge is propagating a PME Message but the PCI-PCI Bridge Function itself is not internally generating a PME.

Figure 7-7 details allocation of register fields for Power Management Capabilities register and Table 7-8 describes the added requirements for this register.

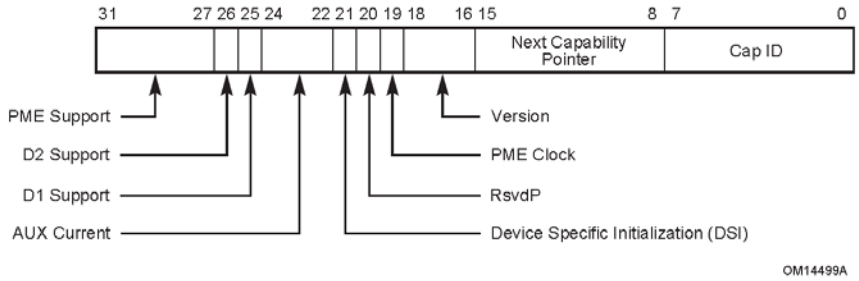


Figure 7-7: Power Management Capabilities Register

Table 7-8: Power Management Capabilities Register Added Requirements

Bit Location	Register Description	Attributes
19	**PME Clock** – Does not apply to PCI Express and must be hardwired to 0b.	Unchanged
31:27	**PME Support** – For a device Function, this 5-bit field indicates the power states in which the Function may generate a PME. Bits 31, 30, and 27 must be Set for PCI-PCI Bridge structures representing Ports on Root Complexes/Switches to indicate that the Bridge will forward PME Messages.	Unchanged

Figure 7-8 details allocation of the register fields for the Power Management Status and Control register and Table 7-9 describes the added requirements for this register.

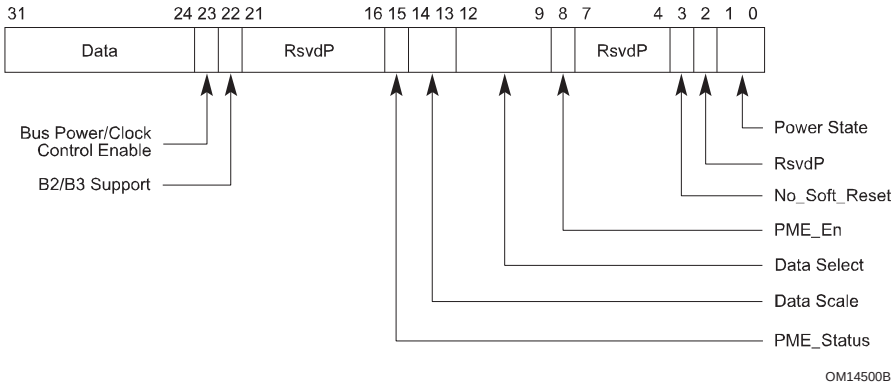


Figure 7-8: Power Management Status/Control Register

Table 7-9: Power Management Status/Control Register Added Requirements

Bit Location	Register Description	Attributes
8	**PME_En** – No added requirements Note: Device Functions that consume Aux power must preserve the value of this sticky register when Aux power is available. In such Functions, this register value is not modified by Conventional Reset or FLR.	Unchanged
15	**PME_Status** – No added requirements Note: Device Functions that consume Aux power must preserve the value of this sticky register when Aux power is available. In such Functions, this register value is not modified by Conventional Reset or FLR.	Unchanged
22	**B2/B3 Support**	Unchanged
23	**Bus Power/Clock Control Enable**	Unchanged

7.7. MSI and MSI-X Capability Structures

All PCI Express device Functions that are capable of generating interrupts must implement MSI or MSI-X or both. MSI, MSI-X, and their Capability structures are defined in the *PCI Local Bus Specification*. The functionality associated with these structures defined by conventional PCI is also required for PCI Express. Only added requirements associated with PCI Express are described here.

7.7.1. Vector Control for MSI-X Table Entries

If a Function implements a TPH Requester Capability structure and an MSI-X Capability structure, the Function can optionally use the Vector Control register in each MSI-X Table Entry to store a Steering Tag. See Section 6.17.

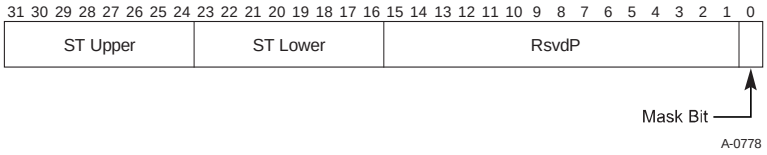


Figure 7-9: Vector Control for MSI-X Table Entries

Table 7-10: Vector Control for MSI-X Table Entries

Bit Location	Register Description	Attributes
0	**Mask Bit** – No added requirements	Unchanged
23:16	**ST Lower** – If the Function implements a TPH Requester Capability structure, and the ST Table Location indicates a value of 10b, then this field contains the lower 8 bits of a Steering Tag. Otherwise, this field is RsvdP. Default value of this field is 00h.	RW
31:24	**ST Upper** – If the Function implements a TPH Requester Capability structure, and the ST Table Location indicates a value of 10b, and the Extended TPH Requester Supported bit is Set, then this field contains the upper 8 bits of a Steering Tag. Otherwise, this field is RsvdP. Default value of this field is 00h.	RW

7.8. PCI Express Capability Structure

PCI Express defines a Capability structure in PCI 3.0 compatible Configuration Space (first 256 bytes) as shown in Figure 7-3. This structure allows identification of a PCI Express device Function and indicates support for new PCI Express features. The PCI Express Capability structure is required for PCI Express device Functions. The Capability structure is a mechanism for enabling PCI software transparent features requiring support on legacy operating systems. In addition to identifying a PCI Express device Function, the PCI Express Capability structure is used to provide access to PCI Express specific Control/Status registers and related Power Management enhancements.

Figure 7-10 details allocation of register fields in the PCI Express Capability structure.

The PCI Express Capabilities, Device Capabilities, Device Status, and Device Control registers are required for all PCI Express device Functions. Device Capabilities 2, Device Status 2, and Device Control 2 registers are required for all PCI Express device Functions that implement capabilities requiring those registers. For device Functions that do not implement the Device Capabilities 2, Device Status 2, and Device Control 2 registers, these spaces must be hardwired to 0b.

The Link Capabilities, Link Status, and Link Control registers are required for all Root Ports, Switch Ports, Bridges, and Endpoints that are not Root Complex Integrated Endpoints. For Functions that do not implement the Link Capabilities, Link Status, and Link Control registers, these spaces must be hardwired to 0. Link Capabilities 2, Link Status 2, and Link Control 2 registers are required for all Root Ports, Switch Ports, Bridges, and Endpoints (except for Root Complex Integrated Endpoints) that implement capabilities requiring those registers. For Functions that do not implement the Link Capabilities 2, Link Status 2, and Link Control 2 registers, these spaces must be hardwired to 0b.

The Slot Capabilities, Slot Status, and Slot Control registers are required in certain~~for~~ Switch Downstream and Root Ports. The Slot Capabilities register is required if ~~a slot is implemented on the Port (indicated by~~ the Slot Implemented bit is Set (see Section 7.8.2)~~in the PCI Express Capabilities register)~~. The Slot Status and Slot Control registers are required if Slot Implemented is Set or if Data Link Layer Link Active Reporting Capable is Set (see Section 7.8.6). Switch Downstream and Root Ports are permitted to implement these registers, even when they are not required. For Functions that do not implement the Slot Capabilities, Slot Status, and Slot Control registers, these spaces must be hardwired to 0b, with the exception of the Presence Detect State bit in the Slot Status register of Downstream Ports, which must be hardwired to 1b (see Section 7.8.11). Slot Capabilities 2, Slot Status 2, and Slot Control 2 registers are required for Switch Downstream and Root Ports if ~~a slot is implemented on the Port and~~ the Function implements capabilities requiring those registers. For Functions that do not implement the Slot Capabilities 2, Slot Status 2, and Slot Control 2 registers, these spaces must be hardwired to 0b.

Comment [VS238]: Errata B39 (Resolved 10/13)

Root Ports and Root Complex Event Collectors must implement the Root Capabilities, Root Status, and Root Control registers. For Functions that do not implement the Root Capabilities, Root Status, and Root Control registers, these spaces must be hardwired to 0b.

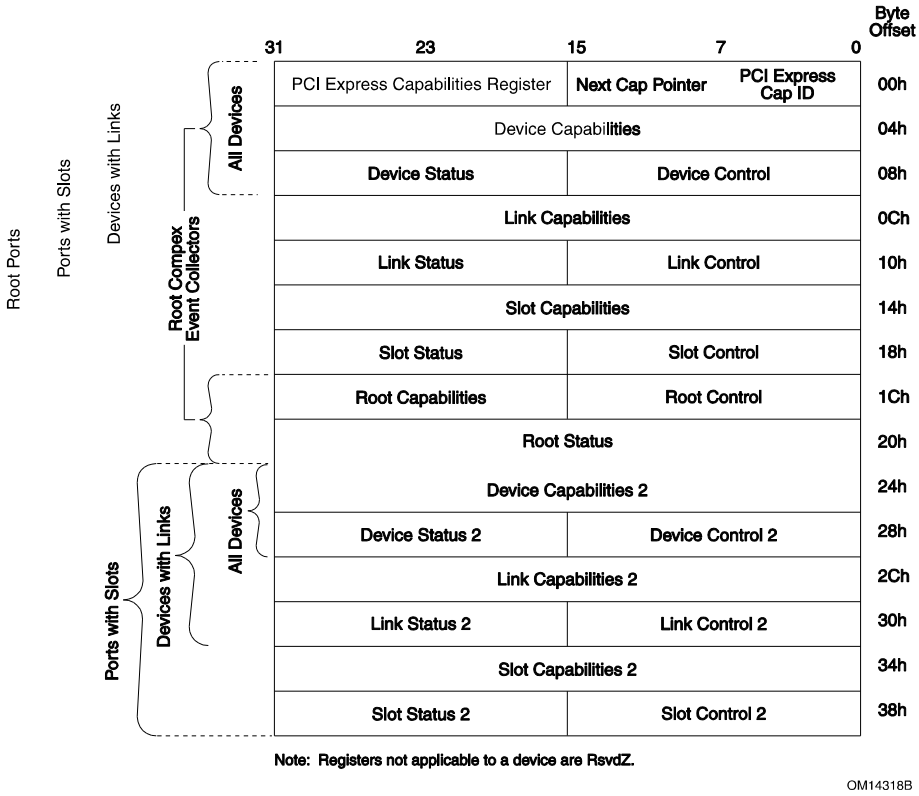


Figure 7-10: PCI Express Capability Structure

7.8.1. PCI Express Capability List Register (Offset 00h)

The PCI Express Capability List register enumerates the PCI Express Capability structure in the PCI 3.0 Configuration Space Capability list. Figure 7-11 details allocation of register fields in the PCI Express Capability List register; Table 7-11 provides the respective bit definitions.

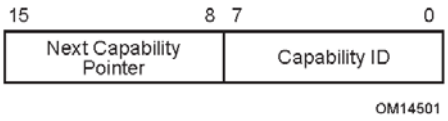


Figure 7-11: PCI Express Capability List Register

Table 7-11: PCI Express Capability List Register

Bit Location	Register Description	Attributes
7:0	**Capability ID** – Indicates the PCI Express Capability structure. This field must return a Capability ID of 10h indicating that this is a PCI Express Capability structure.	RO
15:8	**Next Capability Pointer** – This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of Capabilities.	RO

7.8.2. PCI Express Capabilities Register (Offset 02h)

The PCI Express Capabilities register identifies PCI Express device Function type and associated capabilities. Figure 7-12 details allocation of register fields in the PCI Express Capabilities register; Table 7-12 provides the respective bit definitions.

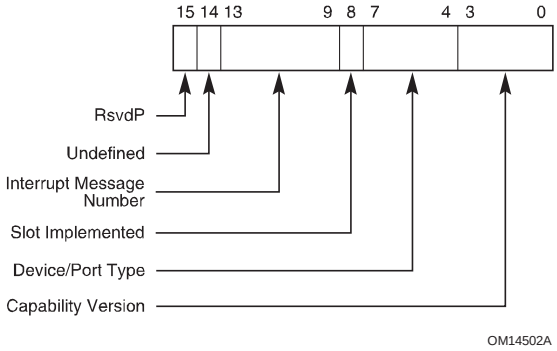


Figure 7-12: PCI Express Capabilities Register

Table 7-12: PCI Express Capabilities Register

Bit Location	Register Description	Attributes
3:0	**Capability Version** – Indicates PCI-SIG defined PCI Express Capability structure version number. A version of the specification that changes the PCI Express Capability structure in a way that is not otherwise identifiable (e.g., through a new Capability field) is permitted to increment this field. All such changes to the PCI Express Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as Functions reporting any such Capability Version numbers will contain a PCI Express Capability structure that is compatible with that piece of software. Must be hardwired to 2h for Functions compliant to this specification.	RO

Bit Location	Register Description	Attributes
7:4	**Device/Port Type** – Indicates the specific type of this PCI Express Function. Note that different Functions in a multi-Function device can generally be of different types. Defined encodings for Functions that implement a Type 00h PCI Configuration Space header are: 0000b PCI Express Endpoint 0001b Legacy PCI Express Endpoint 1001b Root Complex Integrated Endpoint 1010b Root Complex Event Collector Defined encodings for Functions that implement a Type 01h PCI Configuration Space header are: 0100b Root Port of PCI Express Root Complex* 0101b Upstream Port of PCI Express Switch* 0110b Downstream Port of PCI Express Switch* 0111b PCI Express to PCI/PCI-X Bridge* 1000b PCI/PCI-X to PCI Express Bridge* ~~1001b Root Complex Integrated Endpoint~~ ~~1010b Root Complex Event Collector~~ ~~*This value is only valid for Functions that implement a Type 01h PCI Configuration Space header.~~ All other encodings are Reserved. Note that the different Endpoint types have notably different requirements in Section 1.3.2 regarding I/O resources, Extended Configuration Space, and other capabilities.	RO
8	**Slot Implemented** – When Set, this bit indicates that the Link associated with this Port is connected to a slot (as compared to being connected to a system-integrated device or being disabled). This bit is valid for Downstream Ports. This bit is undefined for Upstream Ports.	HwInit

Comment [VS239]: Errata B5

Bit Location	Register Description	Attributes
13:9	**Interrupt Message Number** – This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this Capability structure. For MSI, the value in this field indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the MSI Message Control register. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined.	RO
14	The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate support for TCS Routing. System software should ignore the value read from this bit. System software is permitted to write any value to this bit.	RO

7.8.3. Device Capabilities Register (Offset 04h)

The Device Capabilities register identifies PCI Express device Function specific capabilities. Figure 7-13 details allocation of register fields in the Device Capabilities register; Table 7-13 provides the respective bit definitions.

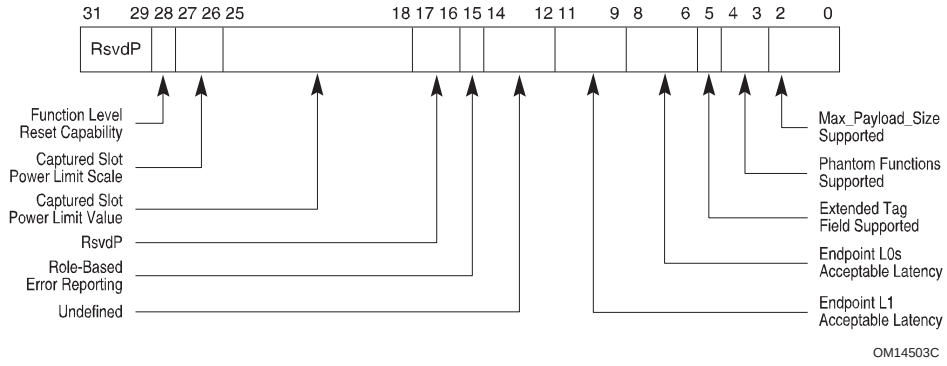


Figure 7-13: Device Capabilities Register

Table 7-13: Device Capabilities Register

Bit Location	Register Description	Attributes
2:0	**Max_Payload_Size Supported** – This field indicates the maximum payload size that the Function can support for TLPs. Defined encodings are: 000b 128 bytes max payload size 001b 256 bytes max payload size 010b 512 bytes max payload size 011b 1024 bytes max payload size 100b 2048 bytes max payload size 101b 4096 bytes max payload size 110b Reserved 111b Reserved The Functions of a multi-Function device are permitted to report different values for this field.	RO

Bit Location	Register Description	Attributes
4:3	**Phantom Functions Supported** – This field indicates the support for use of unclaimed Function Numbers to extend the number of outstanding transactions allowed by logically combining unclaimed Function Numbers (called Phantom Functions) with the Tag identifier (see Section 2.2.6.2 for a description of Tag Extensions). With every Function in an ARI Device, the Phantom Functions Supported field must be set to 00b. The remainder of this field description applies only to non-ARI multi-Function devices. This field indicates the number of most significant bits of the Function Number portion of Requester ID that are logically combined with the Tag identifier. Defined encodings are: 00b – No Function Number bits are used for Phantom Functions. Multi-Function devices are permitted to implement up to 8 independent Functions. 01b – The most significant bit of the Function number in Requester ID is used for Phantom Functions; a multi-Function device is permitted to implement Functions 0-3. Functions 0, 1, 2, and 3 are permitted to use Function Numbers 4, 5, 6, and 7 respectively as Phantom Functions. 10b – The two most significant bits of Function Number in Requester ID are used for Phantom Functions; a multi-Function device is permitted to implement Functions 0-1. Function 0 is permitted to use Function Numbers 2, 4, and 6 for Phantom Functions. Function 1 is permitted to use Function Numbers 3, 5, and 7 as Phantom Functions. 11b – All 3 bits of Function Number in Requester ID used for Phantom Functions. The device must have a single Function 0 that is permitted to use all other Function Numbers as Phantom Functions. Note that Phantom Function support for the Function must be enabled by the Phantom Functions Enable field in the Device Control register before the Function is permitted to use the Function Number field in the Requester ID for Phantom Functions.	RO
5	**Extended Tag Field Supported** – This bit indicates the maximum supported size of the Tag field as a Requester. Defined encodings are: 0b – 5-bit Tag field supported 1b – 8-bit Tag field supported Note that 8-bit Tag field generation must be enabled by the Extended Tag Field Enable bit in the Device Control register of the Requester Function before 8-bit Tags can be generated by the Requester.	RO

Bit Location	Register Description	Attributes
8:6	**Endpoint L0s Acceptable Latency** – This field indicates the acceptable total latency that an Endpoint can withstand due to the transition from L0s state to the L0 state. It is essentially an indirect measure of the Endpoint's internal buffering. Power management software uses the reported L0s Acceptable Latency number to compare against the L0s exit latencies reported by all components comprising the data path from this Endpoint to the Root Complex Root Port to determine whether ASPM L0s entry can be used with no loss of performance. Defined encodings are: 000b Maximum of 64 ns 001b Maximum of 128 ns 010b Maximum of 256 ns 011b Maximum of 512 ns 100b Maximum of 1 µs 101b Maximum of 2 µs 110b Maximum of 4 µs 111b No limit For Functions other than Endpoints, this field is Reserved and must be hardwired to 000b.	RO
11:9	**Endpoint L1 Acceptable Latency** – This field indicates the acceptable latency that an Endpoint can withstand due to the transition from L1 state to the L0 state. It is essentially an indirect measure of the Endpoint's internal buffering. Power management software uses the reported L1 Acceptable Latency number to compare against the L1 Exit Latencies reported (see below) by all components comprising the data path from this Endpoint to the Root Complex Root Port to determine whether ASPM L1 entry can be used with no loss of performance. Defined encodings are: 000b Maximum of 1 µs 001b Maximum of 2 µs 010b Maximum of 4 µs 011b Maximum of 8 µs 100b Maximum of 16 µs 101b Maximum of 32 µs 110b Maximum of 64 µs 111b No limit For Functions other than Endpoints, this field is Reserved and must be hardwired to 000b.	RO

Bit Location	Register Description	Attributes
12	The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate that an Attention Button is implemented on the adapter and electrically controlled by the component on the adapter. System software must ignore the value read from this bit. System software is permitted to write any value to this bit.	RO
13	The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate that an Attention Indicator is implemented on the adapter and electrically controlled by the component on the adapter. System software must ignore the value read from this bit. System software is permitted to write any value to this bit.	RO
14	The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate that a Power Indicator is implemented on the adapter and electrically controlled by the component on the adapter. System software must ignore the value read from this bit. System software is permitted to write any value to this bit.	RO
15	**Role-Based Error Reporting** – When Set, this bit indicates that the Function implements the functionality originally defined in the Error Reporting ECN for *PCI Express Base Specification, Revision 1.0a*, and later incorporated into *PCI Express Base Specification, Revision 1.1*. This bit must be Set by all Functions conforming to the ECN, *PCI Express Base Specification, Revision 1.1*., or subsequent *PCI Express Base Specification* revisions.	RO
25:18	**Captured Slot Power Limit Value** (Upstream Ports only) – In combination with the Captured Slot Power Limit Scale value, specifies the upper limit on power available to the adapter. Power limit (in Watts) is calculated by multiplying the value in this field by the value in the Captured Slot Power Limit Scale field except when the Captured Slot Power Limit Scale field equals 00b (1.0x) and the Captured Slot Power Limit Value exceeds EFh, then alternative encodings are used (see Section 7.8.9). This value is set by the Set_Slot_Power_Limit Message or hardwired to 00h (see Section 6.9). The default value is 00h.	RO
27:26	**Captured Slot Power Limit Scale** (Upstream Ports only) – Specifies the scale used for the Slot Power Limit Value. Range of Values: 00b = 1.0x 01b = 0.1x 10b = 0.01x 11b = 0.001x This value is set by the Set_Slot_Power_Limit Message or hardwired to 00b (see Section 6.9). The default value is 00b.	RO

Bit Location	Register Description	Attributes
28	**Function Level Reset Capability** – A value of 1b indicates the Function supports the optional Function Level Reset mechanism described in Section 6.6.2. This bit applies to Endpoints only. For all other Function types this bit must be hardwired to 0b.	RO

7.8.4. Device Control Register (Offset 08h)

The Device Control register controls PCI Express device specific parameters. Figure 7-14 details allocation of register fields in the Device Control register; Table 7-14 provides the respective bit definitions.

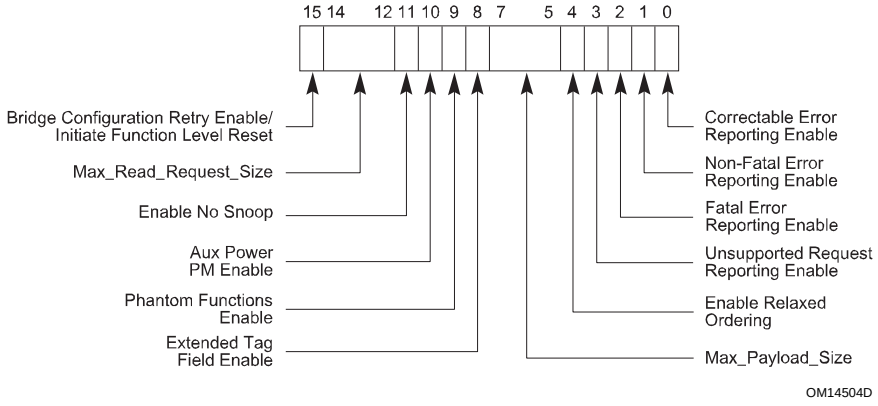


Figure 7-14: Device Control Register

Table 7-14: Device Control Register

Bit Location	Register Description	Attributes
0	**Correctable Error Reporting Enable** – This bit, in conjunction with other bits, controls sending ERR_COR Messages (see Section 6.2.5, and Section 6.2.6, and 6.2.10.2 for details). For a multi-Function device, this bit controls error reporting for each Function from point-of-view of the respective Function. For a Root Port, the reporting of correctable errors is internal to the root. No external ERR_COR Message is generated. A Root Complex Integrated Endpoint that is not associated with a Root Complex Event Collector is permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RW
1	**Non-Fatal Error Reporting Enable** – This bit, in conjunction with other bits, controls sending ERR_NONFATAL Messages (see Section 6.2.5 and Section 6.2.6 for details). For a multi-Function device, this bit controls error reporting for each Function from point-of-view of the respective Function. For a Root Port, the reporting of Non-fatal errors is internal to the root. No external ERR_NONFATAL Message is generated. A Root Complex Integrated Endpoint that is not associated with a Root Complex Event Collector is permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RW
2	**Fatal Error Reporting Enable** – This bit, in conjunction with other bits, controls sending ERR_FATAL Messages (see Section 6.2.5 and Section 6.2.6 for details). For a multi-Function device, this bit controls error reporting for each Function from point-of-view of the respective Function. For a Root Port, the reporting of Fatal errors is internal to the root. No external ERR_FATAL Message is generated. A Root Complex Integrated Endpoint that is not associated with a Root Complex Event Collector is permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RW
3	**Unsupported Request Reporting Enable** – This bit, in conjunction with other bits, controls the signaling of Unsupported Request Errors by sending error Messages (see Section 6.2.5 and Section 6.2.6 for details). For a multi-Function device, this bit controls error reporting for each Function from point-of-view of the respective Function. A Root Complex Integrated Endpoint that is not associated with a Root Complex Event Collector is permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RW

Comment [VS240]: ECN: Downstream Port containment (DPC)

Bit Location	Register Description	Attributes
4	**Enable Relaxed Ordering** – If this bit is Set, the Function is permitted to set the Relaxed Ordering bit in the Attributes field of transactions it initiates that do not require strong write ordering (see Section 2.2.6.4 and Section 2.4). A Function is permitted to hardwire this bit to 0b if it never sets the Relaxed Ordering attribute in transactions it initiates as a Requester. Default value of this bit is 1b.	RW
7:5	**Max_Payload_Size** – This field sets maximum TLP payload size for the Function. As a Receiver, the Function must handle TLPs as large as the set value. As a Transmitter, the Function must not generate TLPs exceeding the set value. Permissible values that can be programmed are indicated by the Max_Payload_Size Supported field in the Device Capabilities register (see Section 7.8.3). Defined encodings for this field are: 000b 128 bytes max payload size 001b 256 bytes max payload size 010b 512 bytes max payload size 011b 1024 bytes max payload size 100b 2048 bytes max payload size 101b 4096 bytes max payload size 110b Reserved 111b Reserved Functions that support only the 128-byte max payload size are permitted to hardwire this field to 000b. System software is not required to program the same value for this field for all the Functions of a multi-Function device. Refer to Section 2.2.2 for important guidance. For ARI Devices, Max_Payload_Size is determined solely by the setting in Function 0. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component. Default value of this field is 000b.	RW
8	**Extended Tag Field Enable** – When Set, this bit enables a Function to use an 8-bit Tag field as a Requester. If the bit is Clear, the Function is restricted to a 5-bit Tag field (see Section 2.2.6.2 for a description of Tag extensions). Functions that do not implement this capability hardwire this bit to 0b. Default value of this bit is implementation specific.	RW

Bit Location	Register Description	Attributes
9	**Phantom Functions Enable** – When Set, this bit enables a Function to use unclaimed Functions as Phantom Functions to extend the number of outstanding transaction identifiers. If the bit is Clear, the Function is not allowed to use Phantom Functions (see Section 2.2.6.2 for a description of Tag extensions). Functions that do not implement this capability hardwire this bit to 0b. Default value of this bit is 0b.	RW
10	**Aux Power PM Enable** – When Set this bit, enables a Function to draw Aux power independent of PME Aux power. Functions that require Aux power on legacy operating systems should continue to indicate PME Aux power requirements. Aux power is allocated as requested in the Aux_Current field of the Power Management Capabilities register (PMC), independent of the PME_En bit in the Power Management Control/Status register (PMCSR) (see Chapter 5). For multi-Function devices, a component is allowed to draw Aux power if at least one of the Functions has this bit set. Note: Functions that consume Aux power must preserve the value of this sticky register when Aux power is available. In such Functions, this bit is not modified by Conventional Reset. Functions that do not implement this capability hardwire this bit to 0b.	RWS
11	**Enable No Snoop** – If this bit is Set, the Function is permitted to Set the No Snoop bit in the Requester Attributes of transactions it initiates that do not require hardware enforced cache coherency (see Section 2.2.6.5). Note that setting this bit to 1b should not cause a Function to Set the No Snoop attribute on all transactions that it initiates. Even when this bit is Set, a Function is only permitted to Set the No Snoop attribute on a transaction when it can guarantee that the address of the transaction is not stored in any cache in the system. This bit is permitted to be hardwired to 0b if a Function would never Set the No Snoop attribute in transactions it initiates. Default value of this bit is 1b.	RW

Bit Location	Register Description	Attributes
14:12	**Max_Read_Request_Size** – This field sets the maximum Read Request size for the Function as a Requester. The Function must not generate Read Requests with a size exceeding the set value. Defined encodings for this field are: 000b 128 bytes maximum Read Request size 001b 256 bytes maximum Read Request size 010b 512 bytes maximum Read Request size 011b 1024 bytes maximum Read Request size 100b 2048 bytes maximum Read Request size 101b 4096 bytes maximum Read Request size 110b Reserved 111b Reserved Functions that do not generate Read Requests larger than 128 bytes and Functions that do not generate Read Requests on their own behalf are permitted to implement this field as Read Only (RO) with a value of 000b. Default value of this field is 010b.	RW
15	*PCI Express to PCI/PCI-X Bridges:* **Bridge Configuration Retry Enable** – When Set, this bit enables PCI Express to PCI/PCI-X bridges to return Configuration Request Retry Status (CRS) in response to Configuration Requests that target devices below the bridge. Refer to the *PCI Express to PCI/PCI-X Bridge Specification, Revision 1.0* for further details. Default value of this bit is 0b. *Endpoints with Function Level Reset Capability set to 1b:* **Initiate Function Level Reset** – A write of 1b initiates Function Level Reset to the Function. The value read by software from this bit is always 0b. *All others:* **Reserved** – Must hardwire the bit to 0b.	*PCI Express to PCI/PCI-X Bridges:* RW *FLR Capable Endpoints:* RW *All others:* RsvdP



IMPLEMENTATION NOTE

Software UR Reporting Compatibility with 1.0a Devices

With 1.0a device Functions,[109] if the Unsupported Request Reporting Enable bit is set, the Function when operating as a Completer will send an uncorrectable error Message (if enabled) when a UR error is detected. On platforms where an uncorrectable error Message is handled as a System Error, this will break PC-compatible Configuration Space probing, so software/firmware on such platforms may need to avoid setting the Unsupported Request Reporting Enable bit.

With device Functions implementing Role-Based Error Reporting, setting the Unsupported Request Reporting Enable bit will not interfere with PC-compatible Configuration Space probing, assuming that the severity for UR is left at its default of non-fatal. However, setting the Unsupported Request Reporting Enable bit will enable the Function to report UR errors[110] detected with posted Requests, helping avoid this case for potential silent data corruption.

On platforms where robust error handling and PC-compatible Configuration Space probing is required, it is suggested that software or firmware have the Unsupported Request Reporting Enable bit Set for Role-Based Error Reporting Functions, but clear for 1.0a Functions. Software or firmware can distinguish the two classes of Functions by examining the Role-Based Error Reporting bit in the Device Capabilities register.

[109] In this context, “1.0a devices” are devices that do not implement Role-Based Error Reporting.

[110] With Role-Based Error Reporting devices, setting the SERR# Enable bit in the Command register also implicitly enables UR reporting.



IMPLEMENTATION NOTE

Use of Max_Payload_Size

The Max_Payload_Size mechanism allows software to control the maximum payload in packets sent by Endpoints to balance latency versus bandwidth trade-offs, particularly for isochronous traffic.

If software chooses to program the Max_Payload_Size of various System Elements to non-default values, it must take care to ensure that each packet does not exceed the Max_Payload_Size parameter of any System Element along the packet's path. Otherwise, the packet will be rejected by the System Element whose Max_Payload_Size parameter is too small.

Discussion of specific algorithms used to configure Max_Payload_Size to meet this requirement is beyond the scope of this specification, but software should base its algorithm upon factors such as the following:

- the Max_Payload_Size capability of each System Element within a hierarchy
- awareness of when System Elements are added or removed through Hot-Plug operations
- knowing which System Elements send packets to each other, what type of traffic is carried, what type of transactions are used, or if packet sizes are constrained by other mechanisms

For the case of firmware that configures System Elements in preparation for running legacy operating system environments, the firmware may need to avoid programming a Max_Payload_Size above the default of 128 bytes, which is the minimum supported by Endpoints.

For example, if the operating system environment does not comprehend PCI Express, firmware probably should not program a non-default Max_Payload_Size for a hierarchy that supports Hot-Plug operations. Otherwise, if no software is present to manage Max_Payload_Size settings when a new element is added, improper operation may result. Note that a newly added element may not even support a Max_Payload_Size setting as large as the rest of the hierarchy, in which case software may need to deny enabling the new element or reduce the Max_Payload_Size settings of other elements.



IMPLEMENTATION NOTE

Use of Max_Read_Request_Size

The Max_Read_Request_Size mechanism allows improved control of bandwidth allocation in systems where quality of service (QoS) is important for the target applications. For example, an arbitration scheme based on counting Requests (and not the sizes of those Requests) provides imprecise bandwidth allocation when some Requesters use much larger sizes than others. The Max_Read_Request_Size mechanism can be used to force more uniform allocation of bandwidth, by restricting the upper size of Read Requests.

7.8.5. Device Status Register (Offset 0Ah)

The Device Status register provides information about PCI Express device (Function) specific parameters. Figure 7-15 details allocation of register fields in the Device Status register; Table 7-15 provides the respective bit definitions.

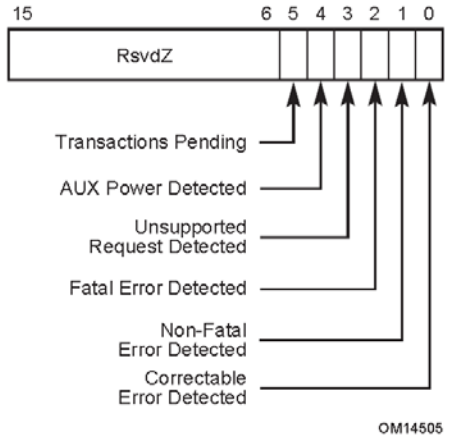


Figure 7-15: Device Status Register

Table 7-15: Device Status Register

Bit Location	Register Description	Attributes
0	**Correctable Error Detected** – This bit indicates status of correctable errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For a multi-Function device, each Function indicates status of errors as perceived by the respective Function. For Functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Correctable Error Mask register. Default value of this bit is 0b.	RW1C
1	**Non-Fatal Error Detected** – This bit indicates status of Non-fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For a multi-Function device, each Function indicates status of errors as perceived by the respective Function. For Functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Uncorrectable Error Mask register. Default value of this bit is 0b.	RW1C

Bit Location	Register Description	Attributes
2	**Fatal Error Detected** – This bit indicates status of Fatal errors detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For a multi-Function device, each Function indicates status of errors as perceived by the respective Function. For Functions supporting Advanced Error Handling, errors are logged in this register regardless of the settings of the Uncorrectable Error Mask register. Default value of this bit is 0b.	RW1C
3	**Unsupported Request Detected** – This bit indicates that the Function received an Unsupported Request. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. For a multi-Function device, each Function indicates status of errors as perceived by the respective Function. Default value of this bit is 0b.	RW1C
4	**AUX Power Detected** – Functions that require Aux power report this bit as Set if Aux power is detected by the Function.	RO
5	**Transactions Pending** – *Endpoints:* When Set, this bit indicates that the Function has issued Non-Posted Requests that have not been completed. A Function reports this bit cleared only when all outstanding Non-Posted Requests have completed or have been terminated by the Completion Timeout mechanism. This bit must also be cleared upon the completion of an FLR. *Root and Switch Ports:* When Set, this bit indicates that a Port has issued Non-Posted Requests on its own behalf (using the Port's own Requester ID) which have not been completed. The Port reports this bit cleared only when all such outstanding Non-Posted Requests have completed or have been terminated by the Completion Timeout mechanism. Note that Root and Switch Ports implementing only the functionality required by this document do not issue Non-Posted Requests on their own behalf, and therefore are not subject to this case. Root and Switch Ports that do not issue Non-Posted Requests on their own behalf hardwire this bit to 0b.	RO

7.8.6. Link Capabilities Register (Offset 0Ch)

The Link Capabilities register identifies PCI Express Link specific capabilities. Figure 7-16 details allocation of register fields in the Link Capabilities register; Table 7-16 provides the respective bit definitions.

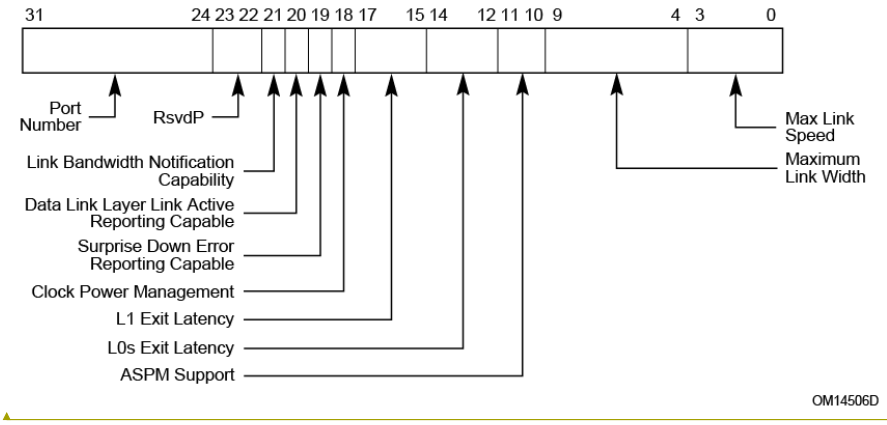


Figure 7-16: Link Capabilities Register

Table 7-16: Link Capabilities Register

Bit Location	Register Description	Attributes
3:0	**Max Link Speed** – This field indicates the maximum Link speed of the associated Port. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the maximum Link speed. Defined encodings are: 0001b Supported Link Speeds Vector field bit 0 0010b Supported Link Speeds Vector field bit 1 0011b Supported Link Speeds Vector field bit 2 0100b Supported Link Speeds Vector field bit 3 0101b Supported Link Speeds Vector field bit 4 0110b Supported Link Speeds Vector field bit 5 0111b Supported Link Speeds Vector field bit 6 All other encodings are reserved. Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions.	RO

Bit Location	Register Description	Attributes
9:4	**Maximum Link Width** – This field indicates the maximum Link width (xN – corresponding to N Lanes) implemented by the component. This value is permitted to exceed the number of Lanes routed to the slot (Downstream Port), adapter connector (Upstream Port), or in the case of component-to-component connections, the actual wired connection width. Defined encodings are: 00 0000b Reserved 00 0001b x1 00 0010b x2 00 0100b x4 00 1000b x8 00 1100b x12 01 0000b x16 10 0000b x32 Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions.	RO
11:10	**Active State Power Management (ASPM) Support** – This field indicates the level of ASPM supported on the given PCI Express Link. See Section 5.4.1 for ASPM support requirements. Defined encodings are: 00b No ASPM Support 01b L0s Supported 10b L1 Supported 11b L0s and L1 Supported Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions.	RO

Bit Location	Register Description	Attributes
14:12	**L0s Exit Latency** – This field indicates the L0s exit latency for the given PCI Express Link. The value reported indicates the length of time this Port requires to complete transition from L0s to L0. If L0s is not supported, the value is undefined; however, see the Implementation Note "Potential Issues With Legacy Software When L0s is Not Supported" in Section 5.4.1.1 for the recommended value. Defined encodings are: 000b Less than 64 ns 001b 64 ns to less than 128 ns 010b 128 ns to less than 256 ns 011b 256 ns to less than 512 ns 100b 512 ns to less than 1 µs 101b 1 µs to less than 2 µs 110b 2 µs-4 µs 111b More than 4 µs Note that exit latencies may be influenced by PCI Express reference clock configuration depending upon whether a component uses a common or separate reference clock. Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions.	RO
17:15	**L1 Exit Latency** – This field indicates the L1 exit latency for the given PCI Express Link. The value reported indicates the length of time this Port requires to complete transition from ASPM L1 to L0. If ASPM L1 is not supported, the value is undefined. Defined encodings are: 000b Less than 1µs 001b 1 µs to less than 2 µs 010b 2 µs to less than 4 µs 011b 4 µs to less than 8 µs 100b 8 µs to less than 16 µs 101b 16 µs to less than 32 µs 110b 32 µs-64 µs 111b More than 64 µs Note that exit latencies may be influenced by PCI Express reference clock configuration depending upon whether a component uses a common or separate reference clock. Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions.	RO

Bit Location	Register Description	Attributes
18	**Clock Power Management** – For Upstream Ports, a value of 1b in this bit indicates that the component tolerates the removal of any reference clock(s) via the "clock request" (CLKREQ#) mechanism when the Link is in the L1 and L2/L3 Ready Link states. A value of 0b indicates the component does not have this capability and that reference clock(s) must not be removed in these Link states. L1 PM Substates defines other semantics for the CLKREQ# signal, which are managed independently of Clock Power Management. This Capability is applicable only in form factors that support "clock request" (CLKREQ#) capability. For a multi-Function device associated with an Upstream Port, each Function indicates its capability independently. Power Management configuration software must only permit reference clock removal if all Functions of the multi-Function device indicate a 1b in this bit. For ARI Devices, all Functions must indicate the same value in this bit. For Downstream Ports, this bit must be hardwired to 0b.	RO
19	**Surprise Down Error Reporting Capable** – For a Downstream Port, this bit must be Set if the component supports the optional capability of detecting and reporting a Surprise Down error condition. For Upstream Ports and components that do not support this optional capability, this bit must be hardwired to 0b.	RO
20	**Data Link Layer Link Active Reporting Capable** – For a Downstream Port, this bit must be hardwired to 1b if the component supports the optional capability of reporting the DL_Active state of the Data Link Control and Management State Machine. For a hot-plug capable Downstream Port (as indicated by the Hot-Plug Capable bit of the Slot Capabilities register) or a Downstream Port that supports Link speeds greater than 5.0 GT/s, this bit must be hardwired to 1b. For Upstream Ports and components that do not support this optional capability, this bit must be hardwired to 0b.	RO
21	**Link Bandwidth Notification Capability** – A value of 1b indicates support for the Link Bandwidth Notification status and interrupt mechanisms. This capability is required for all Root Ports and Switch Downstream Ports supporting Links wider than x1 and/or multiple Link speeds. This field is not applicable and is Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. Functions that do not implement the Link Bandwidth Notification Capability must hardwire this bit to 0b.	RO

Comment [VS241]: ECN: L1 PM Substates with CLKREQ

Bit Location	Register Description	Attributes
22	**ASPM Optionality Compliance** – This bit must be set to 1b in all Functions. Components implemented against certain earlier versions of this specification will have this bit set to 0b. Software is permitted to use the value of this bit to help determine whether to enable ASPM or whether to run ASPM compliance tests.	HwInit
31:24	**Port Number** – This field indicates the PCI Express Port number for the given PCI Express Link. Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions.	HwInit



IMPLEMENTATION NOTE

Use of the ASPM Optionality Compliance Bit

Correct implementation and utilization of ASPM can significantly reduce Link power. However, ASPM feature implementations can be complex, and historically, some implementations have not been compliant to the specification. To address this, some of the ASPM optionality and ASPM entry requirements from earlier revisions of this document have been loosened. However, clear pass/fail compliance testing for ASPM features is also supported and expected.

The ASPM Optionality Compliance bit was created as a tool to establish clear expectations for hardware and software. This bit is Set to indicate hardware that conforms to the current specification, and this bit must be Set in components compliant to this specification.

System software as well as compliance software can assume that if this bit is Set, that the associated hardware conforms to the current specification. Hardware should be fully capable of supporting ASPM configuration management without needing component-specific treatment by system software.

For older hardware that does not have this bit Set, it is strongly recommended for system software to provide mechanisms to enable ASPM on components that work correctly with ASPM, and to disable ASPM on components that don't.

7.8.7. Link Control Register (Offset 10h)

The Link Control register controls PCI Express Link specific parameters. Figure 7-17 details allocation of register fields in the Link Control register; Table 7-17 provides the respective bit definitions.

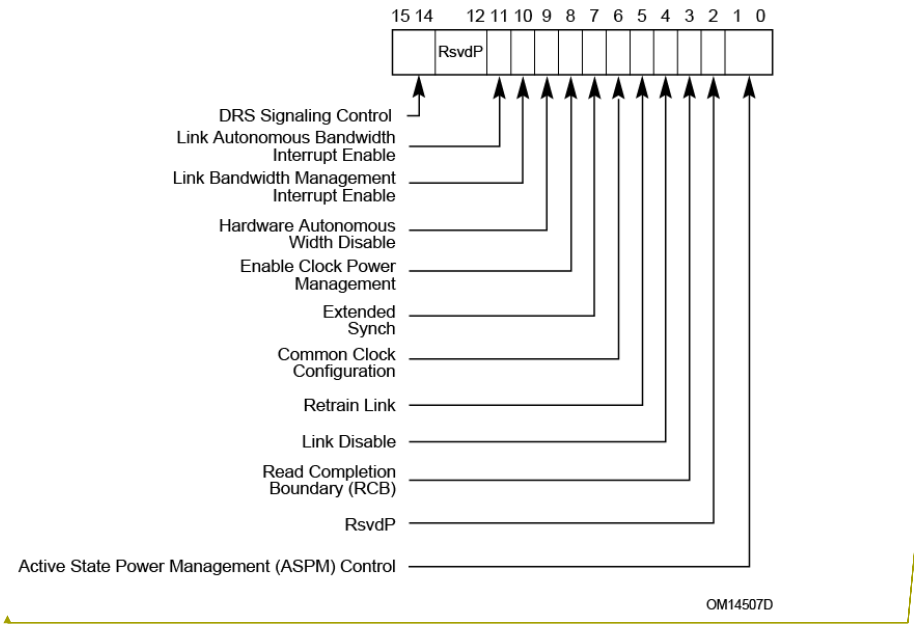


Figure 7-17: Link Control Register

Formatted: Font color: Black

Table 7-17: Link Control Register

Bit Location	Register Description	Attributes
1:0	Active State Power Management (ASPM) Control – This field controls the level of ASPM enabled on the given PCI Express Link. See Section 5.4.1.3 for requirements on when and how to enable ASPM. Defined encodings are: 00b Disabled 01b L0s Entry Enabled 10b L1 Entry Enabled 11b L0s and L1 Entry Enabled Note: "L0s Entry Enabled" enables the Transmitter to enter L0s. If L0s is supported, the Receiver must be capable of entering L0s even when the Transmitter is disabled from entering L0s (00b or 10b). ASPM L1 must be enabled by software in the Upstream component on a Link prior to enabling ASPM L1 in the Downstream component on that Link. When disabling ASPM L1, software must disable ASPM L1 in the Downstream component on a Link prior to disabling ASPM L1 in the Upstream component on that Link. ASPM L1 must only be enabled on the Downstream component if both components on a Link support ASPM L1. For multi-Function devices (including ARI Devices), it is recommended that software program the same value for this field in all Functions. For non-ARI multi-Function devices, only capabilities enabled in all Functions are enabled for the component as a whole. For ARI Devices, ASPM Control is determined solely by the setting in Function 0, regardless of Function 0's D-state. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component. Default value of this field is 00b unless otherwise required by a particular form factor.	RW

Bit Location	Register Description	Attributes
3	Root Ports: Read Completion Boundary (RCB) – Indicates the RCB value for the Root Port. Refer to Section 2.3.1.1 for the definition of the parameter RCB. Defined encodings are: 0b 64 byte 1b 128 byte This bit is hardwired for a Root Port and returns its RCB support capabilities. Endpoints and Bridges: Read Completion Boundary (RCB) – Optionally Set by configuration software to indicate the RCB value of the Root Port Upstream from the Endpoint or Bridge. Refer to Section 2.3.1.1 for the definition of the parameter RCB. Defined encodings are: 0b 64 byte 1b 128 byte Configuration software must only Set this bit if the Root Port Upstream from the Endpoint or Bridge reports an RCB value of 128 bytes (a value of 1b in the Read Completion Boundary bit). Default value of this bit is 0b. Functions that do not implement this feature must hardwire the bit to 0b. Switch Ports: Not applicable – must hardwire the bit to 0b	Root Ports: RO Endpoints and Bridges: RW Switch Ports: RO
4	Link Disable – This bit disables the Link by directing the LTSSM to the Disabled state when Set; this bit is Reserved on Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. Writes to this bit are immediately reflected in the value read from the bit, regardless of actual Link state. After clearing this bit, software must honor timing requirements defined in Section 6.6.1 with respect to the first Configuration Read following a Conventional Reset. Default value of this bit is 0b.	RW

Bit Location	Register Description	Attributes
5	Retrain Link – A write of 1b to this bit initiates Link retraining by directing the Physical Layer LTSSM to the Recovery state. If the LTSSM is already in Recovery or Configuration, re-entering Recovery is permitted but not required. Reads of this bit always return 0b. It is permitted to write 1b to this bit while simultaneously writing modified values to other fields in this register. If the LTSSM is not already in Recovery or Configuration, the resulting Link training must use the modified values. If the LTSSM is already in Recovery or Configuration, the modified values are not required to affect the Link training that's already in progress. This bit is not applicable and is Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. This bit always returns 0b when read.	RW
6	Common Clock Configuration – When Set, this bit indicates that this component and the component at the opposite end of this Link are operating with a distributed common reference clock. A value of 0b indicates that this component and the component at the opposite end of this Link are operating with asynchronous reference clock. For non-ARI multi-Function devices, software must program the same value for this bit in all Functions. If not all Functions are Set, then the component must as a whole assume that its reference clock is not common with the Upstream component. For ARI Devices, Common Clock Configuration is determined solely by the setting in Function 0. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component. Components utilize this common clock configuration information to report the correct L0s and L1 Exit Latencies. After changing the value in this bit in both components on a Link, software must trigger the Link to retrain by writing a 1b to the Retrain Link bit of the Downstream Port. Default value of this bit is 0b.	RW
7	Extended Synch – When Set, this bit forces the transmission of additional Ordered Sets when exiting the L0s state (see Section 4.2.4.5) and when in the Recovery state (see Section 4.2.6.4.1). This mode provides external devices (e.g., logic analyzers) monitoring the Link time to achieve bit and Symbol lock before the Link enters the L0 state and resumes communication. For multi-Function devices if any Function has this bit Set, then the component must transmit the additional Ordered Sets when exiting L0s or when in Recovery. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b. Default value for this bit is 0b.	RW

Comment [VS242]: ECN: M-PCIe

Bit Location	Register Description	Attributes
8	Enable Clock Power Management – Applicable only for Upstream Ports and with form factors that support a "Clock Request" (CLKREQ#) mechanism, this bit operates as follows: 0b Clock power management is disabled and device must hold CLKREQ# signal low. 1b When this bit is Set, the device is permitted to use CLKREQ# signal to power manage Link clock according to protocol defined in appropriate form factor specification. For a non-ARI multi-Function device, power-management-configuration software must only Set this bit if all Functions of the multi-Function device indicate a 1b in the Clock Power Management bit of the Link Capabilities register. The component is permitted to use the CLKREQ# signal to power manage Link clock only if this bit is Set for all Functions. For ARI Devices, Clock Power Management is enabled solely by the setting in Function 0. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component. The CLKREQ# signal may also be controlled via the L1 PM Substates mechanism. Such control is not affected by the setting of this bit. Downstream Ports and components that do not support Clock Power Management (as indicated by a 0b value in the Clock Power Management bit of the Link Capabilities register) must hardwire this bit to 0b. Default value of this bit is 0b, unless specified otherwise by the form factor specification.	RW
9	Hardware Autonomous Width Disable – When Set, this bit disables hardware from changing the Link width for reasons other than attempting to correct unreliable Link operation by reducing Link width. For a Multi-Function device associated with an Upstream Port, the bit in Function 0 is of type RW, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this bit is of type RsvdP. Components that do not implement the ability autonomously to change Link width are permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RW/RsvdP (see description)
10	Link Bandwidth Management Interrupt Enable – When Set, this bit enables the generation of an interrupt to indicate that the Link Bandwidth Management Status bit has been Set. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. Functions that do not implement the Link Bandwidth Notification Capability must hardwire this bit to 0b. Default value of this bit is 0b.	RW

Comment [VS243]: ECN: L1 PM Substates with CLKREQ

Bit Location	Register Description	Attributes
11	Link Autonomous Bandwidth Interrupt Enable – When Set, this bit enables the generation of an interrupt to indicate that the Link Autonomous Bandwidth Status bit has been Set. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. Functions that do not implement the Link Bandwidth Notification Capability must hardwire this bit to 0b. Default value of this bit is 0b.	RW
15:14	DRS Signaling Control – Indicates the mechanism used to report reception of a DRS message. Must be implemented for Downstream Ports with the DRS Supported bit Set in the Link Capabilities 2 Register. Encodings are: 00b DRS not Reported: If DRS Supported is Set, receiving a DRS Message will set DRS Message Received in the Link Status 2 Register but will otherwise have no effect 01b DRS Interrupt Enabled: If the DRS Message Received bit in the Link Status 2 Register transitions from 0 to 1, and either MSI or MSI-X is enabled, an MSI or MSI-X interrupt is generated using the vector in Interrupt Message Number (section 7.8.2) 10b DRS to FRS Signaling Enabled: If the DRS Message Received bit in the Link Status 2 Register transitions from 0 to 1, the Port must send an FRS Message Upstream with the FRS Reason field set to DRS Message Received Behavior is undefined if this field is set to 10b and the FRS Supported bit in the Device Capabilities 2 Register is Clear. Behavior is undefined if this field is set to 11b. Downstream Ports with the DRS Supported bit Clear in the Link Capabilities 2 register must hardwire this field to 00b. This field is Reserved for Upstream Ports. Default value of this field is 00b.	RW/RsvdP



IMPLEMENTATION NOTE

Software Compatibility with ARI Devices

With the ASPM Control field, Common Clock Configuration bit, and Enable Clock Power Management bit in the Link Control register, there are potential software compatibility issues with ARI Devices since these controls operate strictly off the settings in Function 0 instead of the settings in all Functions.

With compliant software, there should be no issues with the Common Clock Configuration bit, since software is required to set this bit the same in all Functions.

With the Enable Clock Power Management bit, there should be no compatibility issues with software that sets this bit the same in all Functions. However, if software does not set this bit the same in all Functions, and relies on each Function having the ability to prevent Clock Power Management from being enabled, such software may have compatibility issues with ARI Devices.

With the ASPM Control field, there should be no compatibility issues with software that sets this bit the same in all Functions. However, if software does not set this bit the same in all Functions, and relies on each Function in D0 state having the ability to prevent ASPM from being enabled, such software may have compatibility issues with ARI Devices.



IMPLEMENTATION NOTE

Avoiding Race Conditions When Using the Retrain Link Bit

When software changes Link control parameters and writes a 1b to the Retrain Link bit in order to initiate Link training using the new parameter settings, special care is required in order to avoid certain race conditions. At any instant the LTSSM may transition to the Recovery or Configuration state due to normal Link activity, without software awareness. If the LTSSM is already in Recovery or Configuration when software writes updated parameters to the Link Control register, as well as a 1b, to the Retrain Link bit, the LTSSM might not use the updated parameter settings with the current Link training, and the current Link training might not achieve the results that software intended.

To avoid this potential race condition, it is highly recommended that software use the following algorithm or something similar:

1. Software sets the relevant Link control parameters to the desired settings without writing a 1b to the Retrain Link bit.
2. Software polls the Link Training bit in the Link Status register until the value returned is 0b.
3. Software writes a 1b to the Retrain Link bit without changing any other fields in the Link Control register.

The above algorithm guarantees that Link training will be based on the Link control parameter settings that software intends.



IMPLEMENTATION NOTE

Use of the Slot Clock Configuration and Common Clock Configuration Bits

In order to determine the common clocking configuration of components on opposite ends of a Link that crosses a connector, two pieces of information are required. The following description defines these requirements.

The first necessary piece of information is whether the Port that connects to the slot uses a clock that has a common source and therefore constant phase relationship to the clock signal provided on the slot. This information is provided by the system side component through a hardware initialized bit (Slot Clock Configuration) in its Link Status register. Note that some electromechanical form factor specifications may require the Port that connects to the slot use a clock that has a common source to the clock signal provided on the slot.

The second necessary piece of information is whether the component on the adapter uses the clock supplied on the slot or one generated locally on the adapter. The adapter design and layout will determine whether the component is connected to the clock source provided by the slot. A component going onto this adapter should have some hardware initialized method for the adapter design/designer to indicate the configuration used for this particular adapter design. This information is reported by bit 12 (Slot Clock Configuration) in the Link Status register of each Function in the Upstream Port. Note that some electromechanical form factor specifications may require the Port on the adapter to use the clock signal provided on the connector.

System firmware or software will read this value from the components on both ends of a physical Link. If both components report the use of a common clock connection this firmware/software will program bit 6 (Common Clock Configuration) of the Link Control register to 1b on both components connected to the Link. Each component uses this bit to determine the length of time required to re-synch its Receiver to the opposing component's Transmitter when exiting L0s.

This value is reported as a time value in bits 12-14 of the Link Capabilities register (offset 0Ch) and is sent to the opposing Transmitter as part of the initialization process as N_FTS. Components would be expected to require much longer synch times without common clocking and would therefore report a longer L0s exit latency in bits 12-14 of the Link Capabilities register and would send a larger number for N_FTS during training. This forces a requirement that whatever software changes this bit should force a Link retrain in order to get the correct N-FTS set for the Receivers at both ends of the Link.

7.8.8. Link Status Register (Offset 12h)

The Link Status register provides information about PCI Express Link specific parameters. Figure 7-18 details allocation of register fields in the Link Status register; Table 7-18 provides the respective bit definitions.

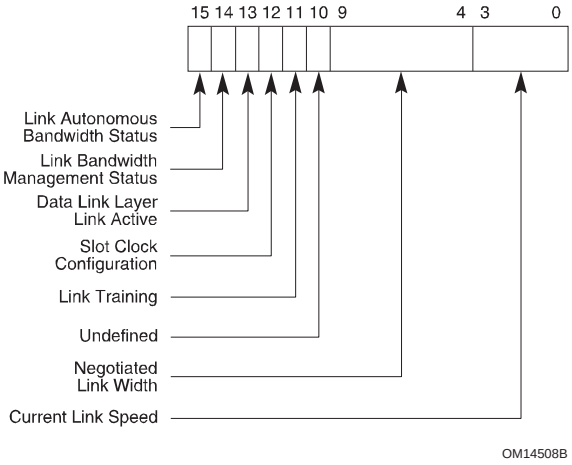


Figure 7-18: Link Status Register

Table 7-18: Link Status Register

Bit Location	Register Description	Attributes
3:0	**Current Link Speed** – This field indicates the negotiated Link speed of the given PCI Express Link. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the current Link speed. Defined encodings are: 0001b Supported Link Speeds Vector field bit 0 0010b Supported Link Speeds Vector field bit 1 0011b Supported Link Speeds Vector field bit 2 0100b Supported Link Speeds Vector field bit 3 0101b Supported Link Speeds Vector field bit 4 0110b Supported Link Speeds Vector field bit 5 0111b Supported Link Speeds Vector field bit 6 All other encodings are Reserved. The value in this field is undefined when the Link is not up.	RO

Bit Location	Register Description	Attributes
9:4	**Negotiated Link Width** – This field indicates the negotiated width of the given PCI Express Link. Defined encodings are: 00 0001b x1 00 0010b x2 00 0100b x4 00 1000b x8 00 1100b x12 01 0000b x16 10 0000b x32 All other encodings are Reserved. The value in this field is undefined when the Link is not up.	RO
10	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate a Link Training Error. System software must ignore the value read from this bit. System software is permitted to write any value to this bit.	RO
11	**Link Training** – This read-only bit indicates that the Physical Layer LTSSM is in the Configuration or Recovery state, or that 1b was written to the Retrain Link bit but Link training has not yet begun. Hardware clears this bit when the LTSSM exits the Configuration/Recovery state. This bit is not applicable and Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches, and must be hardwired to 0b.	RO
12	**Slot Clock Configuration** – This bit indicates that the component uses the same physical reference clock that the platform provides on the connector. If the device uses an independent clock irrespective of the presence of a reference clock on the connector, this bit must be clear. For a multi-Function device, each Function must report the same value for this bit.	HwInit
13	**Data Link Layer Link Active** – This bit indicates the status of the Data Link Control and Management State Machine. It returns a 1b to indicate the DL_Active state, 0b otherwise. This bit must be implemented if the Data Link Layer Link Active Reporting Capable bit is 1b. Otherwise, this bit must be hardwired to 0b.	RO

Bit Location	Register Description	Attributes
14	**Link Bandwidth Management Status** – This bit is Set by hardware to indicate that either of the following has occurred without the Port transitioning through DL_Down status: • A Link retraining has completed following a write of 1b to the Retrain Link bit. Note: This bit is Set following any write of 1b to the Retrain Link bit, including when the Link is in the process of retraining for some other reason. • Hardware has changed Link speed or width to attempt to correct unreliable Link operation, either through an LTSSM timeout or a higher level process. This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was not indicated as an autonomous change. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. Functions that do not implement the Link Bandwidth Notification Capability must hardwire this bit to 0b. The default value of this bit is 0b.	RW1C
15	**Link Autonomous Bandwidth Status** – This bit is Set by hardware to indicate that hardware has autonomously changed Link speed or width, without the Port transitioning through DL_Down status, for reasons other than to attempt to correct unreliable Link operation. This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was indicated as an autonomous change. This bit is not applicable and is Reserved for Endpoints, PCI Express-to-PCI/PCI-X bridges, and Upstream Ports of Switches. Functions that do not implement the Link Bandwidth Notification Capability must hardwire this bit to 0b. The default value of this bit is 0b.	RW1C

7.8.9. Slot Capabilities Register (Offset 14h)

The Slot Capabilities register identifies PCI Express slot specific capabilities. Figure 7-19 details allocation of register fields in the Slot Capabilities register; Table 7-19 provides the respective bit definitions.

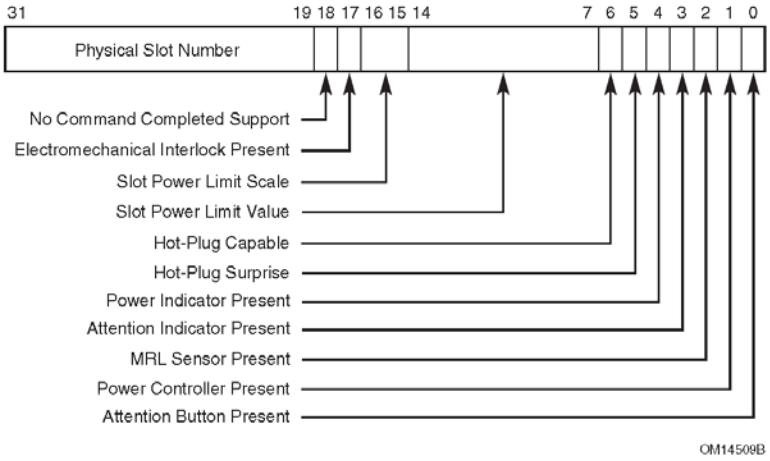


Figure 7-19: Slot Capabilities Register

Table 7-19: Slot Capabilities Register

Bit Location	Register Description	Attributes
0	**Attention Button Present** – When Set, this bit indicates that an Attention Button for this slot is electrically controlled by the chassis.	HwInit
1	**Power Controller Present** – When Set, this bit indicates that a software programmable Power Controller is implemented for this slot/adapter (depending on form factor).	HwInit
2	**MRL Sensor Present** – When Set, this bit indicates that an MRL Sensor is implemented on the chassis for this slot.	HwInit
3	**Attention Indicator Present** – When Set, this bit indicates that an Attention Indicator is electrically controlled by the chassis.	HwInit
4	**Power Indicator Present** – When Set, this bit indicates that a Power Indicator is electrically controlled by the chassis for this slot.	HwInit
5	**Hot-Plug Surprise** – When Set, this bit indicates that an adapter present in this slot might be removed from the system without any prior notification. This is a form factor specific capability. This bit is an indication to the operating system to allow for such removal without impacting continued software operation.	HwInit

Bit Location	Register Description	Attributes
6	**Hot-Plug Capable** – When Set, this bit indicates that this slot is capable of supporting hot-plug operations.	HwInit
14:7	**Slot Power Limit Value** – In combination with the Slot Power Limit Scale value, specifies the upper limit on power supplied by the slot (see Section 6.9) or by other means to the adapter. Power limit (in Watts) is calculated by multiplying the value in this field by the value in the Slot Power Limit Scale field except when the Slot Power Limit Scale field equals 00b (1.0x) and Slot Power Limit Value exceeds EFh, the following alternative encodings are used: F0h = 250 W Slot Power Limit F1h = 275 W Slot Power Limit F2h = 300 W Slot Power Limit F3h to FFh = Reserved for Slot Power Limit values above 300 W This register must be implemented if the Slot Implemented bit is Set. Writes to this register also cause the Port to send the Set_Slot_Power_Limit Message. The default value prior to hardware/firmware initialization is 0000 0000b.	HwInit
16:15	**Slot Power Limit Scale** – Specifies the scale used for the Slot Power Limit Value (see Section 6.9). Range of Values: 00b = 1.0x 01b = 0.1x 10b = 0.01x 11b = 0.001x This register must be implemented if the Slot Implemented bit is Set. Writes to this register also cause the Port to send the Set_Slot_Power_Limit Message. The default value prior to hardware/firmware initialization is 00b.	HwInit
17	**Electromechanical Interlock Present** – When Set, this bit indicates that an Electromechanical Interlock is implemented on the chassis for this slot.	HwInit
18	**No Command Completed Support** – When Set, this bit indicates that this slot does not generate software notification when an issued command is completed by the Hot-Plug Controller. This bit is only permitted to be Set if the hot-plug capable Port is able to accept writes to all fields of the Slot Control register without delay between successive writes.	HwInit

Bit Location	Register Description	Attributes
31:19	**Physical Slot Number** – This field indicates the physical slot number attached to this Port. This field must be hardware initialized to a value that assigns a slot number that is unique within the chassis, regardless of the form factor associated with the slot. This field must be initialized to zero for Ports connected to devices that are either integrated on the system board or integrated within the same silicon as the Switch device or Root Port.	HwInit

7.8.10. Slot Control Register (Offset 18h)

The Slot Control register controls PCI Express Slot specific parameters. Figure 7-20 details allocation of register fields in the Slot Control register; Table 7-20 provides the respective bit definitions.

Attention Indicator Control, Power Indicator Control, and Power Controller Control fields of the Slot Control register do not have a defined default value. If these fields are implemented, it is the responsibility of either system firmware or operating system software to (re)initialize these fields after a reset of the Link.

In hot-plug capable Downstream Ports, a write to the Slot Control register must cause a hot-plug command to be generated (see Section 6.7.3.2 for details on hot-plug commands). A write to the Slot Control register in a Downstream Port that is not hot-plug capable must not cause a hot-plug command to be executed.

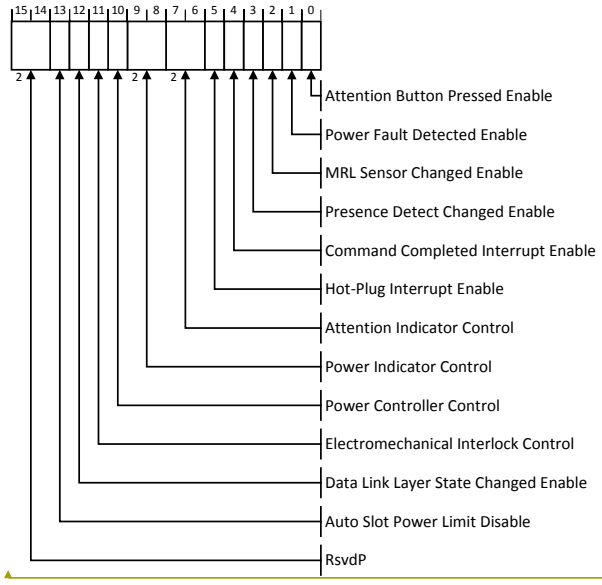


Figure 7-20: Slot Control Register

Formatted: Font color: Black

Comment [VS244]: ECN: Downstream Port containment (DPC)

Table 7-20: Slot Control Register

Bit Location	Register Description	Attributes
0	**Attention Button Pressed Enable** – When Set to 1b, this bit enables software notification on an attention button pressed event (see Section 6.7.3). If the Attention Button Present bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 0b. Default value of this bit is 0b.	RW
1	**Power Fault Detected Enable** – When Set, this bit enables software notification on a power fault event (see Section 6.7.3). If a Power Controller that supports power fault detection is not implemented, this bit is permitted to be read-only with a value of 0b. Default value of this bit is 0b.	RW
2	**MRL Sensor Changed Enable** – When Set, this bit enables software notification on a MRL sensor changed event (see Section 6.7.3). If the MRL Sensor Present bit in the Slot Capabilities register is Clear, this bit is permitted to be read-only with a value of 0b. Default value of this bit is 0b.	RW
3	**Presence Detect Changed Enable** – When Set, this bit enables software notification on a presence detect changed event (see Section 6.7.3). If the Hot-Plug Capable bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 0b. Default value of this bit is 0b.	RW
4	**Command Completed Interrupt Enable** – If Command Completed notification is supported (if the No Command Completed Support bit in the Slot Capabilities register is 0b), when Set, this bit enables software notification when a hot-plug command is completed by the Hot-Plug Controller. If Command Completed notification is not supported, this bit must be hardwired to 0b. Default value of this bit is 0b.	RW
5	**Hot-Plug Interrupt Enable** – When Set, this bit enables generation of an interrupt on enabled hot-plug events. If the Hot Plug Capable bit in the Slot Capabilities register is Clear, this bit is permitted to be read-only with a value of 0b. Default value of this bit is 0b.	RW

Bit Location	Register Description	Attributes
7:6	**Attention Indicator Control** – If an Attention Indicator is implemented, writes to this field set the Attention Indicator to the written state. Reads of this field must reflect the value from the latest write, even if the corresponding hot-plug command is not complete, unless software issues a write without waiting, if required to, for the previous command to complete in which case the read value is undefined. Defined encodings are: 00b Reserved 01b On 10b Blink 11b Off Note: The default value of this field must be one of the non-Reserved values. If the Attention Indicator Present bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 00b.	RW
9:8	**Power Indicator Control** – If a Power Indicator is implemented, writes to this field set the Power Indicator to the written state. Reads of this field must reflect the value from the latest write, even if the corresponding hot-plug command is not complete, unless software issues a write without waiting, if required to, for the previous command to complete in which case the read value is undefined. Defined encodings are: 00b Reserved 01b On 10b Blink 11b Off Note: The default value of this field must be one of the non-Reserved values. If the Power Indicator Present bit in the Slot Capabilities register is 0b, this bit is permitted to be read-only with a value of 00b.	RW

Bit Location	Register Description	Attributes
10	**Power Controller Control** – If a Power Controller is implemented, this bit when written sets the power state of the slot per the defined encodings. Reads of this bit must reflect the value from the latest write, even if the corresponding hot-plug command is not complete, unless software issues a write, if required to, without waiting for the previous command to complete in which case the read value is undefined. Note that in some cases the power controller may autonomously remove slot power or not respond to a power-up request based on a detected fault condition, independent of the Power Controller Control setting. The defined encodings are: 0b Power On 1b Power Off If the Power Controller Present bit in the Slot Capabilities register is Clear, then writes to this bit have no effect and the read value of this bit is undefined.	RW
11	**Electromechanical Interlock Control** – If an Electromechanical Interlock is implemented, a write of 1b to this bit causes the state of the interlock to toggle. A write of 0b to this bit has no effect. A read of this bit always returns a 0b.	RW
12	**Data Link Layer State Changed Enable** – If the Data Link Layer Link Active Reporting capability is 1b, this bit enables software notification when Data Link Layer Link Active bit is changed. If the Data Link Layer Link Active Reporting Capable bit is 0b, this bit is permitted to be read-only with a value of 0b. Default value of this bit is 0b.	RW
13	**Auto Slot Power Limit Disable** – When Set, this disables the automatic sending of a Set_Slot_Power_Limit Message when a Link transitions from a non-DL_Up status to a DL_Up status. Downstream ports that don't support DPC are permitted to hardwire this bit to 0. Default value of this bit is implementation specific.	RW

Comment [VS246]: ECN: Downstream Port containment (DPC)

Comment [VS245]: Errata B31 (Part C list resolved 10/13)

7.8.11. Slot Status Register (Offset 1Ah)

The Slot Status register provides information about PCI Express Slot specific parameters. Figure 7-21 details allocation of register fields in the Slot Status register; Table 7-21 provides the respective bit definitions. Register fields for status bits not implemented by the device have the RsvdZ attribute.

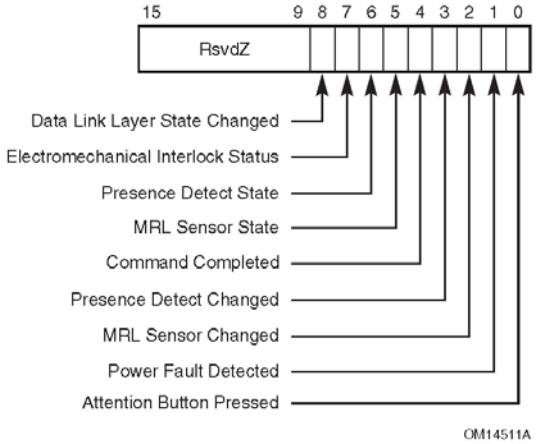


Figure 7-21: Slot Status Register

Table 7-21: Slot Status Register

Bit Location	Register Description	Attributes
0	**Attention Button Pressed** – If an Attention Button is implemented, this bit is Set when the attention button is pressed. If an Attention Button is not supported, this bit must not be Set.	RW1C
1	**Power Fault Detected** – If a Power Controller that supports power fault detection is implemented, this bit is Set when the Power Controller detects a power fault at this slot. Note that, depending on hardware capability, it is possible that a power fault can be detected at any time, independent of the Power Controller Control setting or the occupancy of the slot. If power fault detection is not supported, this bit must not be Set.	RW1C
2	**MRL Sensor Changed** – If an MRL sensor is implemented, this bit is Set when a MRL Sensor state change is detected. If an MRL sensor is not implemented, this bit must not be Set.	RW1C
3	**Presence Detect Changed** – This bit is set when the value reported in the Presence Detect State bit is changed.	RW1C

Bit Location	Register Description	Attributes
4	**Command Completed** – If Command Completed notification is supported (if the No Command Completed Support bit in the Slot Capabilities register is 0b), this bit is Set when a hot-plug command has completed and the Hot-Plug Controller is ready to accept a subsequent command. The Command Completed status bit is Set as an indication to host software that the Hot-Plug Controller has processed the previous command and is ready to receive the next command; it provides no guarantee that the action corresponding to the command is complete. If Command Completed notification is not supported, this bit must be hardwired to 0b.	RW1C
5	**MRL Sensor State** – This bit reports the status of the MRL sensor if implemented. Defined encodings are: 0b MRL Closed 1b MRL Open	RO
6	**Presence Detect State** – This bit indicates the presence of an adapter in the slot, reflected by the logical "OR" of the Physical Layer in-band presence detect mechanism and, if present, any out-of-band presence detect mechanism defined for the slot's corresponding form factor. Note that the in-band presence detect mechanism requires that power be applied to an adapter for its presence to be detected. Consequently, form factors that require a power controller for hot-plug must implement a physical pin presence detect mechanism. Defined encodings are: 0b Slot Empty 1b Card Present in slot This bit must be implemented on all Downstream Ports that implement slots. For Downstream Ports not connected to slots (where the Slot Implemented bit of the PCI Express Capabilities register is 0b), this bit must be hardwired to 1b.	RO
7	**Electromechanical Interlock Status** – If an Electromechanical Interlock is implemented, this bit indicates the status of the Electromechanical Interlock. Defined encodings are: 0b Electromechanical Interlock Disengaged 1b Electromechanical Interlock Engaged	RO
8	**Data Link Layer State Changed** – This bit is Set when the value reported in the Data Link Layer Link Active bit of the Link Status register is changed. In response to a Data Link Layer State Changed event, software must read the Data Link Layer Link Active bit of the Link Status register to determine if the Link is active before initiating configuration cycles to the hot plugged device.	RW1C

IMPLEMENTATION NOTE

No Slot Power Controller

For slots that do not implement a power controller, software must ensure that system power planes are enabled to provide power to slots prior to reading presence detect state.

7.8.12. Root Control Register (Offset 1Ch)

The Root Control register controls PCI Express Root Complex specific parameters. Figure 7-22 details allocation of register fields in the Root Control register; Table 7-22 provides the respective bit definitions.

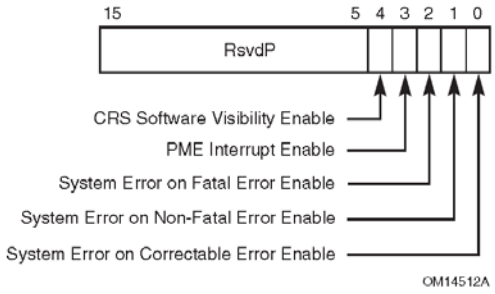


Figure 7-22: Root Control Register

Table 7-22: Root Control Register

Bit Location	Register Description	Attributes
0	**System Error on Correctable Error Enable** – If Set, this bit indicates that a System Error should be generated if a correctable error (ERR_COR) is reported by any of the devices in the ~~hierarchy~~ Hierarchy Domain associated with this Root Port, or by the Root Port itself. The mechanism for signaling a System Error to the system is system specific. Root Complex Event Collectors provide support for the above-described functionality for Root Complex Integrated Endpoints. Default value of this bit is 0b.	RW
1	**System Error on Non-Fatal Error Enable** – If Set, this bit indicates that a System Error should be generated if a Non-fatal error (ERR_NONFATAL) is reported by any of the devices in the ~~hierarchy~~ Hierarchy Domain associated with this Root Port, or by the Root Port itself. The mechanism for signaling a System Error to the system is system specific. Root Complex Event Collectors provide support for the above-described functionality for Root Complex Integrated Endpoints. Default value of this bit is 0b.	RW

Comment [VS247]: Errata A20

Comment [VS248]: A20

Bit Location	Register Description	Attributes
2	**System Error on Fatal Error Enable** – If Set, this bit indicates that a System Error should be generated if a Fatal error (ERR_FATAL) is reported by any of the devices in the ~~hierarchy~~ Hierarchy Domain associated with this Root Port, or by the Root Port itself. The mechanism for signaling a System Error to the system is system specific. Root Complex Event Collectors provide support for the above-described functionality for Root Complex Integrated Endpoints. Default value of this bit is 0b.	RW
3	**PME Interrupt Enable** – When Set, this bit enables PME interrupt generation upon receipt of a PME Message as reflected in the PME Status bit (see Table 7-24). A PME interrupt is also generated if the PME Status bit is Set when this bit is changed from Clear to Set (see Section 6.1.5). Default value of this bit is 0b.	RW
4	**CRS Software Visibility Enable** – When Set, this bit enables the Root Port to return Configuration Request Retry Status (CRS) Completion Status to software (see Section 2.3.1). Root Ports that do not implement this capability must hardwire this bit to 0b. Default value of this bit is 0b.	RW

Comment [VS249]: A20

7.8.13. Root Capabilities Register (Offset 1Eh)

The Root Capabilities register identifies PCI Express Root Port specific capabilities. Figure 7-23 details allocation of register fields in the Root Capabilities register; Table 7-23 provides the respective bit definitions.

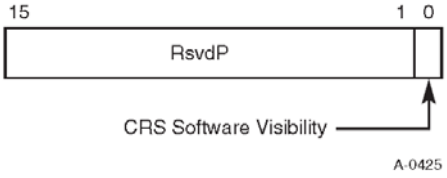


Figure 7-23: Root Capabilities Register

Table 7-23: Root Capabilities Register

Bit Location	Register Description	Attributes
0	**CRS Software Visibility** – When Set, this bit indicates that the Root Port is capable of returning Configuration Request Retry Status (CRS) Completion Status to software (see Section 2.3.1).	RO

7.8.14. Root Status Register (Offset 20h)

The Root Status register provides information about PCI Express device specific parameters. Figure 7-24 details allocation of register fields in the Root Status register; Table 7-24 provides the respective bit definitions.

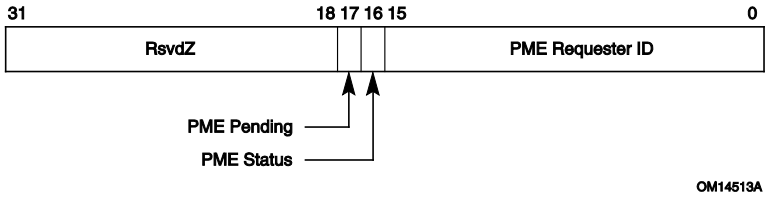


Figure 7-24: Root Status Register

Table 7-24: Root Status Register

Bit Location	Register Description	Attributes
15:0	**PME Requester ID** – This field indicates the PCI Requester ID of the last PME Requester. This field is only valid when the PME Status bit is Set.	RO
16	**PME Status** – This bit indicates that PME was asserted by the PME Requester indicated in the PME Requester ID field. Subsequent PMEs are kept pending until the status register is cleared by software by writing a 1b. Default value of this bit is 0b.	RW1C
17	**PME Pending** – This bit indicates that another PME is pending when the PME Status bit is Set. When the PME Status bit is cleared by software; the PME is delivered by hardware by setting the PME Status bit again and updating the PME Requester ID field appropriately. The PME Pending bit is cleared by hardware if no more PMEs are pending.	RO

7.8.15. Device Capabilities 2 Register (Offset 24h)

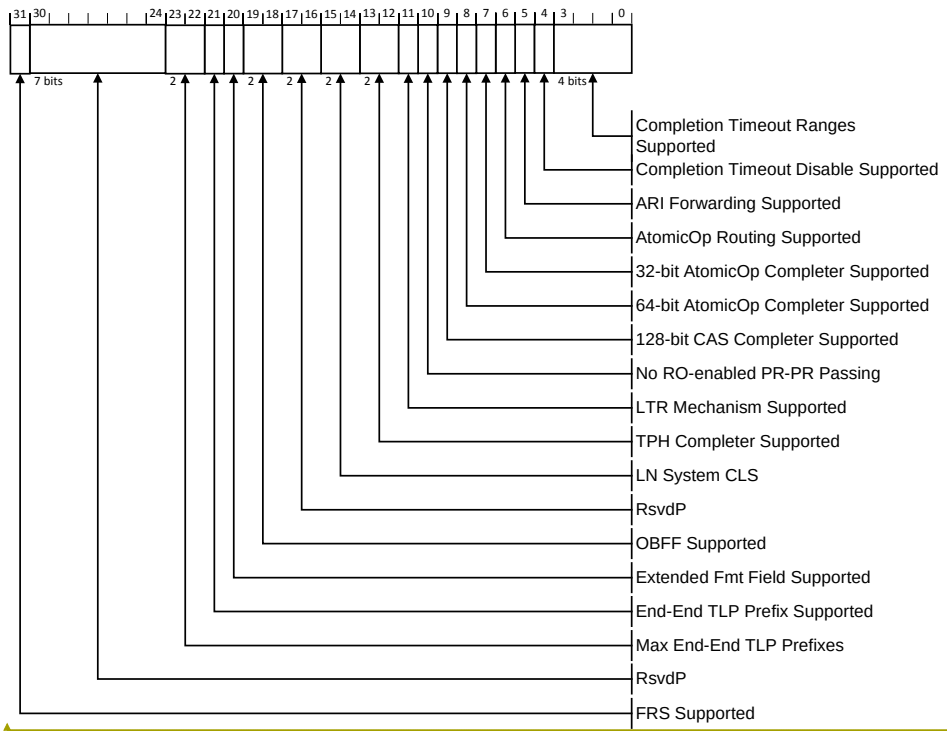


Figure 7-25: Device Capabilities 2 Register

Formatted: Font: Font color: Black

Comment [VS250]: Bits 15:14 added per ECN: Lightweight Notification (LN) Protocol

Comment [VS251]: Bits 17:16 and 15:14 added by LN ECN

Comment [VS252]: Bit 31 added per D-ECN-Readiness Notifications

Table 7-25: Device Capabilities 2 Register

Bit Location	Register Description	Attributes
3:0	**Completion Timeout Ranges Supported** – This field indicates device Function support for the optional Completion Timeout programmability mechanism. This mechanism allows system software to modify the Completion Timeout value. This field is applicable only to Root Ports, Endpoints that issue Requests on their own behalf, and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. For all other Functions this field is Reserved and must be hardwired to 0000b. Four time value ranges are defined: Range A: 50 µs to 10 ms Range B: 10 ms to 250 ms Range C: 250 ms to 4 s Range D: 4 s to 64 s Bits are set according to the table below to show timeout value ranges supported. 0000b Completion Timeout programming not supported – the Function must implement a timeout value in the range 50 µs to 50 ms. 0001b Range A 0010b Range B 0011b Ranges A and B 0110b Ranges B and C 0111b Ranges A, B, and C 1110b Ranges B, C, and D 1111b Ranges A, B, C, and D All other values are Reserved. It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms.	HwInit

Bit Location	Register Description	Attributes
4	**Completion Timeout Disable Supported** – A value of 1b indicates support for the Completion Timeout Disable mechanism. The Completion Timeout Disable mechanism is required for Endpoints that issue Requests on their own behalf and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. This mechanism is optional for Root Ports. For all other Functions this field is Reserved and must be hardwired to 0b.	RO
5	**ARI Forwarding Supported** – Applicable only to Switch Downstream Ports and Root Ports; must be 0b for other Function types. This bit must be set to 1b if a Switch Downstream Port or Root Port supports this optional capability. See Section 6.13 for additional details.	RO
6	**AtomicOp Routing Supported** – Applicable only to Switch Upstream Ports, Switch Downstream Ports, and Root Ports; must be 0b for other Function types. This bit must be set to 1b if the Port supports this optional capability. See Section 6.15 for additional details.	RO
7	**32-bit AtomicOp Completer Supported** – Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. See Section 6.15.3.1 for additional RC requirements.	RO
8	**64-bit AtomicOp Completer Supported** – Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. See Section 6.15.3.1 for additional RC requirements.	RO
9	**128-bit CAS Completer Supported** – Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. This bit must be set to 1b if the Function supports this optional capability. See Section 6.15 for additional details.	RO
10	**No RO-enabled PR-PR Passing** – If this bit is Set, the routing element never carries out the passing permitted by Table 2-39 entry A2b that is associated with the Relaxed Ordering Attribute field being Set. This bit applies only for Switches and RCs that support peer-to-peer traffic between Root Ports. This bit applies only to Posted Requests being forwarded through the Switch or RC and does not apply to traffic originating or terminating within the Switch or RC itself. All Ports on a Switch or RC must report the same value for this bit. For all other functions, this bit must be 0b.	HwInit

Bit Location	Register Description	Attributes
11	**LTR Mechanism Supported** – A value of 1b indicates support for the optional Latency Tolerance Reporting (LTR) mechanism. Root Ports, Switches and Endpoints are permitted to implement this capability. For a multi-Function device associated with an Upstream Port, each Function must report the same value for this bit. For Bridges and other Functions that do not implement this capability, this bit must be hardwired to 0b.	RO
13:12	**TPH Completer Supported** – Value indicates Completer support for TPH or Extended TPH. Applicable only to Root Ports and Endpoints. For all other Functions, this field is Reserved. Defined Encodings are: 00b TPH and Extended TPH Completer not supported. 01b TPH Completer supported; Extended TPH Completer not supported. 10b Reserved. 11b Both TPH and Extended TPH Completer supported. See Section 6.17 for details.	RO
15:14	**LN System CLS** – Applicable only to Root Ports and RCRBs; must be 00b for all other Function types. This field indicates if the Root Port or RCRB supports LN protocol as an LN Completer, and if so, what cacheline size is in effect. Encodings are: 00b LN Completer either not supported or not in effect 01b LN Completer with 64-byte cachelines in effect 10b LN Completer with 128-byte cachelines in effect 11b Reserved	HWInit

Comment [VS253]: ECN: Lightweight Notification (LN) Protocol

Bit Location	Register Description	Attributes
19:18	**OBFF Supported** – This field indicates if OBFF is supported and, if so, what signaling mechanism is used. 00b OBFF Not Supported 01b OBFF supported using Message signaling only 10b OBFF supported using WAKE# signaling only 11b OBFF supported using WAKE# and Message signaling The value reported in this field must indicate support for WAKE# signaling only if: - for a Downstream Port, driving the WAKE# signal for OBFF is supported and the connector or component connected Downstream is known to receive that same WAKE# signal - for an Upstream Port, receiving the WAKE# signal for OBFF is supported and, if the component is on an add-in-card, that the component is connected to the WAKE# signal on the connector. Root Ports, Switch Ports, and Endpoints are permitted to implement this capability. For a multi-Function device associated with an Upstream Port, each Function must report the same value for this field. For Bridges and Ports that do not implement this capability, this field must be hardwired to 00b.	Hwinit
20	**Extended Fmt Field Supported** – If Set, the Function supports the 3-bit definition of the Fmt field. If Clear, the Function supports a 2-bit definition of the Fmt field. See Section 2.2. Must be Set for Functions that support End-End TLP Prefixes. All Functions in an Upstream Port must have the same value for this bit. Each Downstream Port of a component may have a different value for this bit. It is strongly recommended that Functions support the 3-bit definition of the Fmt field.	RO
21	**End-End TLP Prefix Supported** – Indicates whether End-End TLP Prefix support is offered by a Function. Values are: 0b No Support 1b Support is provided to receive TLPs containing End-End TLP Prefixes. All Ports of a Switch must have the same value for this bit**.**	HwInit

Bit Location	Register Description	Attributes
23:22	**Max End-End TLP Prefixes –** Indicates the maximum number of End-End TLP Prefixes supported by this Function. See Section 2.2.10.2 for important details. Values are: 01b 1 End-End TLP Prefix 10b 2 End-End TLP Prefixes 11b 3 End-End TLP Prefixes 00b 4 End-End TLP Prefixes If End-End TLP Prefix Supported is Clear, this field is RsvdP. Different Root Ports that have the End-End TLP Prefix Supported bit Set are permitted to report different values for this field. For Switches where End-End TLP Prefix Supported is Set, this field must be 00b indicating support for up to four End-End TLP Prefixes.	HwInit
31	**FRS Supported** – When Set, indicates support for the optional Function Readiness Status (FRS) capability. Must be Set for all Functions that support generation or receipt capabilities of FRS Messages. Must not be Set by Switch Functions that do not generate FRS Messages on their own behalf.	HwInit

Comment [VS254]: Updated per RC_ECN_RN_27_Aug_2013



IMPLEMENTATION NOTE

Use of the No RO-enabled PR-PR Passing Bit

The No RO-enabled PR-PR Passing bit allows platforms to utilize PCI Express switching elements on the path between a requester and completer for requesters that could benefit from a slightly less relaxed ordering model. An example is a device that cannot ensure that multiple overlapping posted writes to the same address are outstanding at the same time. The method by which such a device is enabled to utilize this mode is beyond the scope of this specification.

7.8.16. Device Control 2 Register (Offset 28h)

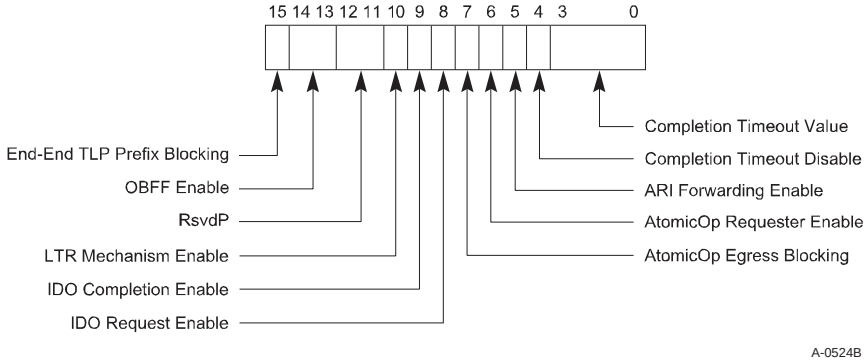


Figure 7-26: Device Control 2 Register

Table 7-26: Device Control 2 Register

Bit Location	Register Description	Attributes
3:0	**Completion Timeout Value** – In device Functions that support Completion Timeout programmability, this field allows system software to modify the Completion Timeout value. This field is applicable to Root Ports, Endpoints that issue Requests on their own behalf, and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. For all other Functions this field is Reserved and must be hardwired to 0000b. A Function that does not support this optional capability must hardwire this field to 0000b and is required to implement a timeout value in the range 50 µs to 50 ms. Functions that support Completion Timeout programmability must support the values given below corresponding to the programmability ranges indicated in the Completion Timeout Ranges Supported field. Defined encodings: 0000b Default range: 50 µs to 50 ms It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms. Values available if Range A (50 µs to 10 ms) programmability range is supported: 0001b 50 µs to 100 µs 0010b 1 ms to 10 ms Values available if Range B (10 ms to 250 ms) programmability range is supported: 0101b 16 ms to 55 ms 0110b 65 ms to 210 ms Values available if Range C (250 ms to 4 s) programmability range is supported: 1001b 260 ms to 900 ms 1010b 1 s to 3.5 s Values available if the Range D (4 s to 64 s) programmability range is supported: 1101b 4 s to 13 s 1110b 17 s to 64 s Values not defined above are Reserved. Software is permitted to change the value in this field at any time. For Requests already pending when the Completion Timeout Value is changed, hardware is permitted to use either the new or the old value for the outstanding Requests, and is permitted to base the start time for each Request either on when this value was changed or on when each request was issued. The default value for this field is 0000b.	RW

Bit Location	Register Description	Attributes
4	**Completion Timeout Disable** – When Set, this bit disables the Completion Timeout mechanism. This bit is required for all Functions that support the Completion Timeout Disable Capability. Functions that do not support this optional capability are permitted to hardwire this bit to 0b Software is permitted to Set or Clear this bit at any time. When Set, the Completion Timeout detection mechanism is disabled. If there are outstanding Requests when the bit is cleared, it is permitted but not required for hardware to apply the completion timeout mechanism to the outstanding Requests. If this is done, it is permitted to base the start time for each Request on either the time this bit was cleared or the time each Request was issued. The default value for this bit is 0b.	RW
5	**ARI Forwarding Enable** – When set, the Downstream Port disables its traditional Device Number field being 0 enforcement when turning a Type 1 Configuration Request into a Type 0 Configuration Request, permitting access to Extended Functions in an ARI Device immediately below the Port. See Section 6.13. Default value of this bit is 0b. Must be hardwired to 0b if the ARI Forwarding Supported bit is 0b.	RW
6	**AtomicOp Requester Enable** – Applicable only to Endpoints and Root Ports; must be hardwired to 0b for other Function types. The Function is allowed to initiate AtomicOp Requests only if this bit and the Bus Master Enable bit in the Command register are both Set. This bit is required to be RW if the Endpoint or Root Port is capable of initiating AtomicOp Requests, but otherwise is permitted to be hardwired to 0b. This bit does not serve as a capability bit. This bit is permitted to be RW even if no AtomicOp Requester capabilities are supported by the Endpoint or Root Port. Default value of this bit is 0b.	RW
7	**AtomicOp Egress Blocking** – Applicable and mandatory for Switch Upstream Ports, Switch Downstream Ports, and Root Ports that implement AtomicOp routing capability; otherwise must be hardwired to 0b. When this bit is Set, AtomicOp Requests that target going out this Egress Port must be blocked. See Section 6.15.2. Default value of this bit is 0b.	RW
8	**IDO Request Enable** – If this bit is Set, the Function is permitted to set the ID-Based Ordering (IDO) bit (Attr[2]) of Requests it initiates (see Section 2.2.6.3 and Section 2.4). Endpoints, including RC Integrated Endpoints, and Root Ports are permitted to implement this capability. A Function is permitted to hardwire this bit to 0b if it never sets the IDO attribute in Requests. Default value of this bit is 0b.	RW

Bit Location	Register Description	Attributes
9	**IDO Completion Enable** – If this bit is Set, the Function is permitted to set the ID-Based Ordering (IDO) bit (Attr[2]) of Completions it returns (see Section 2.2.6.3 and Section 2.4). Endpoints, including RC Integrated Endpoints, and Root Ports are permitted to implement this capability. A Function is permitted to hardwire this bit to 0b if it never sets the IDO attribute in Completions. Default value of this bit is 0b.	RW
10	**LTR Mechanism Enable** – When Set to 1b, this bit enables Upstream Ports to send LTR messages and Downstream Ports to process LTR Messages. For a Multi-Function device associated with an Upstream Port of a device that implements LTR, the bit in Function 0 is RW, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this bit is RsvdP. Functions that do not implement the LTR mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b. For Downstream Ports, this bit must be reset to the default value if the Port goes to DL_Down status.	RW/RsvdP
14:13	**OBFF Enable** – This field enables the OBFF mechanism and selects the signaling method. 00b Disabled 01b Enabled using Message signaling [Variation A] 10b Enabled using Message signaling [Variation B] 11b Enabled using WAKE# signaling See Section 6.19 for an explanation of the above encodings. This field is required for all Ports that support the OBFF Capability. For a Multi-Function Device associated with an Upstream Port of a Device that implements OBFF, the field in Function 0 is of type RW, and only Function 0 controls the Component's behavior. In all other Functions of that Device, this field is of type RsvdP. Ports that do not implement OBFF are permitted to hardwire this field to 00b. Default value of this field is 00b.	RW/RsvdP (see description)

Bit Location	Register Description	Attributes
15	**End-End TLP Prefix Blocking** – Controls whether the routing function is permitted to forward TLPs containing an End-End TLP Prefix. Values are: 0b Forwarding Enabled – Function is permitted to send TLPs with End-End TLP Prefixes. 1b Forwarding Blocked – Function is not permitted to send TLPs with End-End TLP Prefixes. This bit affects TLPs that exit the Switch/Root Complex using the associated Port. It does not affect TLPs forwarded internally within the Switch/Root Complex. It does not affect TLPs that enter through the associated Port, that originate in the associated Port or originate in a Root Complex Integrated Device integrated with the associated Port. As described in Section 2.2.10.2, blocked TLPs are reported by the TLP Prefix Blocked Error. The default value of this bit is 0b. This bit is hardwired to 1b in Root Ports that support End-End TLP Prefixes but do not support forwarding of End-End TLP Prefixes. This bit is applicable to Root Ports and Switch Ports where the End-End TLP Prefix Supported bit is Set. This bit is not applicable and is RsvdP in all other cases.	RW (see description)

7.8.17. Device Status 2 Register (Offset 2Ah)

This section is a placeholder. There are no capabilities that require this register.

This register must be treated by software as RsvdZ.

7.8.18. Link Capabilities 2 Register (Offset 2Ch)

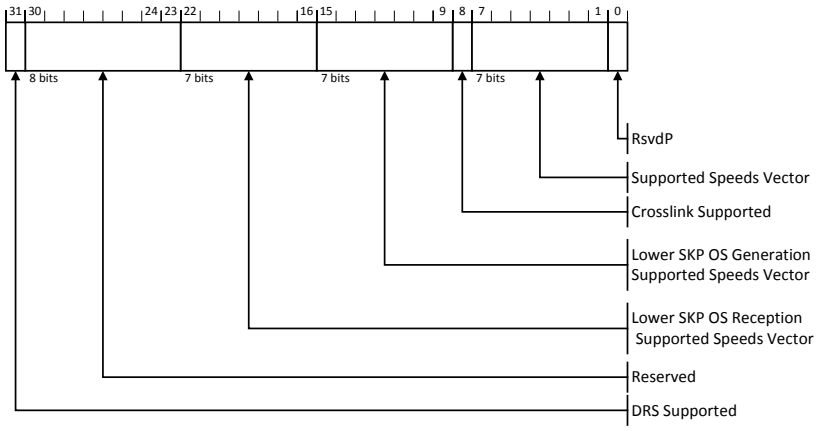


Figure 7-27: Link Capabilities 2 Register

Comment [VS255]: ECN:SRIS

Table 7-27: Link Capabilities 2 Register

Bit Location	Register Description	Attributes
7:1	**Supported Link Speeds Vector** – This field indicates the supported Link speed(s) of the associated Port. For each bit, a value of 1b indicates that the corresponding Link speed is supported; otherwise, the Link speed is not supported. See Section 4.3.2 for further requirements. Bit definitions within this field are: Bit 0 2.5 GT/s Bit 1 5.0 GT/s Bit 2 8.0 GT/s Bits 6:3 RsvdP Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions. For M-PCIe the bit definitions within this field are: Bit 0 HS-G1/HS-G2 Bit 1 HS-G3 Bits 6:2 RsvdP	RO
8	**Crosslink Supported** – When set to 1b, this bit indicates that the associated Port supports crosslinks (see Section 4.2.6.3.1). When set to 0b on a Port that supports Link speeds of 8.0 GT/s or higher, this bit indicates that the associated Port does not support crosslinks. When set to 0b on a Port that only supports Link speeds of 2.5 GT/s or 5.0 GT/s, this bit provides no information regarding the Port's level of crosslink support. It is recommended that this bit be Set in any Port that supports crosslinks even though doing so is only required for Ports that also support operating at 8.0 GT/s or higher Link speeds. Note: Software should use this bit when referencing fields whose definition depends on whether or not the Port supports crosslinks (see Section 7.27.4). Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions. With the M-PCIe PHY, Crosslinks are not supported and this bit must be hardwired to 0b.	RO

Comment [VS256]: Errata A1

Comment [VS257]: ECN: M-PCIe

Comment [VS258]: ECN: M-PCIe

Bit Location	Register Description	Attributes
9:15	**Lower SKP OS Generation Supported Speeds Vector** – If this field is non-zero, it indicates that the Port, when operating at the indicated speed(s) supports SRIS and also supports software control of the SKP Ordered Set transmission scheduling rate. Bit definitions within this field are: Bit 0 2.5 GT/s Bit 1 5.0 GT/s Bit 2 8.0 GT/s Bits 6:3 RsvdP Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions. Behavior is undefined if a bit is Set in this field and the corresponding bit is not Set in the Supported Link Speeds Vector.	HwInit/RsvdP
16:22	**Lower SKP OS Reception Supported Speeds Vector** – If this field is non-zero, it indicates that the Port, when operating at the indicated speed(s) supports SRIS and also supports receiving SKP OS at the rate defined for SRNS while running in SRIS. Bit definitions within this field are: Bit 0 2.5 GT/s Bit 1 5.0 GT/s Bit 2 8.0 GT/s Bits 6:3 RsvdP Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions. Behavior is undefined if a bit is Set in this field and the corresponding bit is not Set in the Supported Link Speeds Vector.	HwInit/RsvdP
31	**DRS Supported** – When Set, indicates support for the optional Device Readiness Status (DRS) capability. Must be Set in Downstream Ports that support DRS. Must be Set in Downstream Ports that support FRS. For Upstream Ports that support DRS, it is strongly recommended that this bit be Set in Function 0. For all other Functions associated with an Upstream Port, this bit must be Clear.[111] Must be Clear in Functions that are not associated with a Port. RsvdP in all other Functions.	HwInit/RsvdP~~/RO~~

Comment [VS259]: ECN:SRIS

Comment [VS260]: ECN:SRIS

Comment [VS261]: Overridden to HwInit/RsvdP during the Aug-Sept update. (D-ECN-Readiness Notifications had RO)

[111] It is expressly permitted for Upstream Ports to send DRS Messages even when the DRS Supported bit is Clear.

IMPLEMENTATION NOTE

Software Management of Link Speeds With Earlier Hardware

Hardware components compliant to versions of the Base Specification prior to Revision 3.0 either did not implement the Link Capabilities 2 register, or the register was Reserved.

For software to determine the supported Link speeds for components where the Link Capabilities 2 register is either not implemented, or the value of its Supported Link Speeds Vector is 0000000b, software can read bits 3:0 of the Link Capabilities register (now defined to be the Max Link Speed field), and interpret the value as follows:

0001b 2.5 GT/s Link speed supported

0010b 5.0 GT/s and 2.5 GT/s Link speeds supported

For such components, the encoding of the values for the Current Link Speed field (in the Link Status register) and Target Link Speed field (in the Link Control 2 register) is the same as above.

IMPLEMENTATION NOTE

Software Management of Link Speeds With Future Hardware

It is strongly encouraged that software primarily utilize the Supported Link Speeds Vector instead of the Max Link Speed field, so that software can determine the exact set of supported speeds on current and future hardware. This can avoid software being confused if a future specification defines Links that do not require support for all slower speeds.

Formatted: Font: Arial Black, 14 pt, Font color: Black, All caps

Comment [VS262]: Errata A2

7.8.19. Link Control 2 Register (Offset 30h)

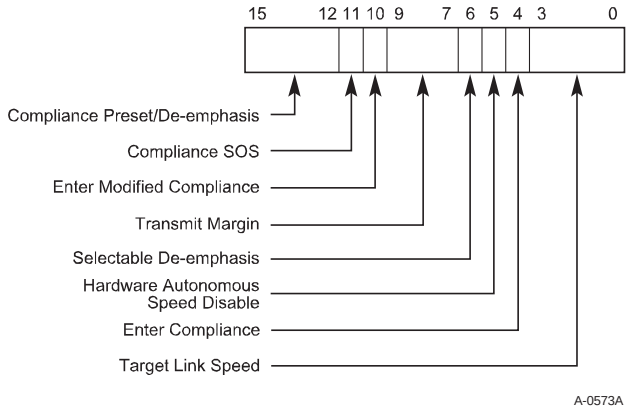


Figure 7-28: Link Control 2 Register

Table 7-28: Link Control 2 Register

Bit Location	Register Description	Attributes
3:0	**Target Link Speed** – For Downstream Ports, this field sets an upper limit on Link operational speed by restricting the values advertised by the Upstream component in its training sequences. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the desired target Link speed. Defined encodings are: 0001b Supported Link Speeds Vector field bit 0 0010b Supported Link Speeds Vector field bit 1 0011b Supported Link Speeds Vector field bit 2 0100b Supported Link Speeds Vector field bit 3 0101b Supported Link Speeds Vector field bit 4 0110b Supported Link Speeds Vector field bit 5 0111b Supported Link Speeds Vector field bit 6 All other encodings are Reserved. If a value is written to this field that does not correspond to a supported speed (as indicated by the Supported Link Speeds Vector), the result is undefined. If either of the Enter Compliance or Enter Modified Compliance bits are implemented, then this field must also be implemented. The default value of this field is the highest Link speed supported by the component (as reported in the Max Link Speed field of the Link Capabilities register) unless the corresponding platform/form factor requires a different default value. For both Upstream and Downstream Ports, this field is used to set the target compliance mode speed when software is using the Enter Compliance bit to force a Link into compliance mode. For a Multi-Function device associated with an Upstream Port, the field in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this field is of type RsvdP. Components that support only the 2.5 GT/s speed are permitted to hardwire this field to 0000b.	RWS/RsvdP (see description)

Bit Location	Register Description	Attributes
4	**Enter Compliance** – Software is permitted to force a Link to enter Compliance mode (at the speed indicated in the Target Link Speed field and the de-emphasis/preset level indicated by the Compliance Preset/De-emphasis field) by setting this bit to 1b in both components on a Link and then initiating a hot reset on the Link. Default value of this bit following Fundamental Reset is 0b. For a Multi-Function device associated with an Upstream Port, the bit in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this bit is of type RsvdP. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this bit only during debug or compliance testing. In all other cases, the system must ensure that this bit is set to the default value.	RWS/RsvdP (see description)
5	**Hardware Autonomous Speed Disable** – When Set, this bit disables hardware from changing the Link speed for device-specific reasons other than attempting to correct unreliable Link operation by reducing Link speed. Initial transition to the highest supported common link speed is not blocked by this bit. For a Multi-Function device associated with an Upstream Port, the bit in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this bit is of type RsvdP. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RWS/RsvdP (see description)
6	**Selectable De-emphasis** – When the Link is operating at 5.0 GT/s speed, this bit is used to control the transmit de-emphasis of the link in specific situations. See Section 4.2.6 for detailed usage information. Encodings: 1b -3.5 dB 0b -6 dB When the Link is not operating at 5.0 GT/s speed, the setting of this bit has no effect. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. This bit is not applicable and Reserved for Endpoints, PCI Express to PCI/PCI-X bridges, and Upstream Ports of Switches. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	HwInit

Comment [VS263]: ECN: M-PCIe

Bit Location	Register Description	Attributes
9:7	**Transmit Margin** – This field controls the value of the non-deemphasized voltage level at the Transmitter pins. This field is reset to 000b on entry to the LTSSM Polling.Configuration substate (see Chapter 4 for details of how the Transmitter voltage level is determined in various states). Encodings: 000b Normal operating range 001b-111b As defined in Section 4.3.3.6, not all encodings are required to be implemented. For a Multi-Function device associated with an Upstream Port, the field in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this field is of type RsvdP. Default value of this field is 000b. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 000b. This field is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this field only during debug or compliance testing. In all other cases, the system must ensure that this field is set to the default value. With the M-PCIe PHY, this feature is unused and this field is hardwired to 000b.	RWS/RsvdP (see description)
10	**Enter Modified Compliance** – When this bit is set to 1b, the device transmits Modified Compliance Pattern if the LTSSM enters Polling.Compliance substate. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. For a Multi-Function device associated with an Upstream Port, the bit in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this bit is of type RsvdP. Default value of this bit is 0b. This bit is intended for debug, compliance testing purposes only. System firmware and software is allowed to modify this bit only during debug or compliance testing. In all other cases, the system must ensure that this bit is set to the default value. With the M-PCIe PHY, this feature is unused and this field is hardwired to 0b.	RWS/RsvdP (see description)

Comment [VS264]: ECN: M-PCIe

Comment [VS265]: ECN: M-PCIe

Bit Location	Register Description	Attributes
11	**Compliance SOS** – When set to 1b, the LTSSM is required to send SKP Ordered Sets between sequences when sending the Compliance Pattern or Modified Compliance Pattern. For a Multi-Function device associated with an Upstream Port, the bit in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this bit is of type RsvdP. The default value of this bit is 0b. This bit is applicable when the Link is operating at 2.5 GT/s or 5.0 GT/s data rates only. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	RWS/RsvdP (see description)
15:12	**Compliance Preset/De-emphasis** – For 8.0 GT/s Data Rate: This field sets the Transmitter Preset in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b. The encodings are defined in Section 4.2.3.2. Results are undefined if a reserved preset encoding is used when entering Polling.Compliance in this way. For 5.0 GT/s Data Rate: This field sets the de-emphasis level in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b. Defined Encodings are: 0001b -3.5 dB 0000b -6 dB When the Link is operating at 2.5 GT/s, the setting of this field has no effect. Components that support only 2.5 GT/s speed are permitted to hardwire this field to 0000b. For a Multi-Function device associated with an Upstream Port, the field in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this field is of type RsvdP. The default value of this field is 0000b. This field is intended for debug and compliance testing purposes. System firmware and software is allowed to modify this field only during debug or compliance testing. In all other cases, the system must ensure that this field is set to the default value. With the M-PCIe PHY, this feature is unused and this field is hardwired to 0000b.	RWS/RsvdP (see description)

Comment [VS266]: ECN: M-PCIe

Comment [VS267]: ECN: M-PCIe

IMPLEMENTATION NOTE

Selectable De-emphasis Usage

Selectable De-emphasis setting is applicable only to Root Ports and Switch Downstream Ports. The De-emphasis setting is implementation specific and depends on the platform or enclosure in which the Root Port or the Switch Downstream Port is located. System firmware or hardware strapping is used to configure the selectable de-emphasis value. In cases where system firmware cannot be used to set the de-emphasis value (for example, a hot plugged Switch), hardware strapping must be used to set the de-emphasis value.

7.8.20. Link Status 2 Register (Offset 32h)

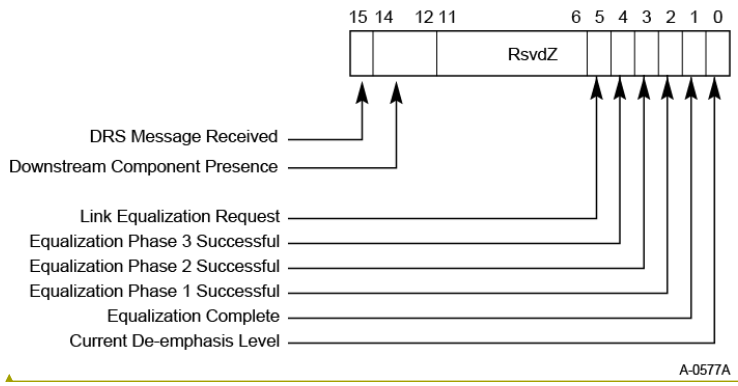


Figure 7-29: Link Status 2 Register

Formatted: Font:

Comment [VS268]: Bits 15, 14:12 added per D-ECN-Readiness Notifications

Table 7-29: Link Status 2 Register

Bit Location	Register Description	Attributes
0	**Current De-emphasis Level** – When the Link is operating at 5.0 GT/s speed, this bit reflects the level of de-emphasis. Encodings: 1b -3.5 dB 0b -6 dB The value in this bit is undefined when the Link is not operating at 5.0 GT/s speed. Components that support only the 2.5 GT/s speed are permitted to hardwire this bit to 0b. For components that support speeds greater than 2.5 GT/s, Multi-Function devices associated with an Upstream Port must report the same value in this field for all Functions of the Port. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	RO
1	**Equalization Complete** – When set to 1b, this bit indicates that the Transmitter Equalization procedure has completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in Section 4.2.6.4.2. The default value of this bit is 0b. For Multi-Function Upstream Port, this bit must be implemented in Function 0 and RsvdZ in other Functions. Components that only support speeds below 8.0 GT/s are permitted to hardwire this bit to 0b. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	ROS
2	**Equalization Phase 1 Successful** – When set to 1b, this bit indicates that Phase 1 of the Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in Section 4.2.6.4.2. The default value of this bit is 0b. For Multi-Function Upstream Port, this bit must be implemented in Function 0 and RsvdZ in other Functions. Components that only support speeds below 8.0 GT/s are permitted to hardwire this bit to 0b. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	ROS

Comment [VS269]: ECN: M-PCIe

Comment [VS270]: ECN: M-PCIe

Comment [VS271]: ECN: M-PCIe

Bit Location	Register Description	Attributes
3	**Equalization Phase 2 Successful** – When set to 1b, this bit indicates that Phase 2 of the Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in Section 4.2.6.4.2. The default value of this bit is 0b. For Multi-Function Upstream Port, this bit must be implemented in Function 0 and RsvdZ in other Functions. Components that only support speeds below 8.0 GT/s are permitted to hardwire this bit to 0b. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	ROS
4	**Equalization Phase 3 Successful** – When set to 1b, this bit indicates that Phase 3 of the Transmitter Equalization procedure has successfully completed. Details of the Transmitter Equalization process and when this bit needs to be set to 1b is provided in Section 4.2.6.4.2. The default value of this bit is 0b. For Multi-Function Upstream Port, this bit must be implemented in Function 0 and RsvdZ in other Functions. Components that only support speeds below 8.0 GT/s are permitted to hardwire this bit to 0b. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	ROS
5	**Link Equalization Request** – This bit is Set by hardware to request the Link equalization process to be performed on the Link. Refer to Sections 4.2.3 and 4.2.6.4.2 for details. The default value of this bit is 0b. For Multi-Function Upstream Port, this bit must be implemented in Function 0 and RsvdZ in other Functions. Components that only support speeds below 8.0 GT/s are permitted to hardwire this bit to 0b. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b.	RW1CS

Comment [VS272]: ECN: M-PCIe

Comment [VS273]: ECN: M-PCIe

Comment [VS274]: ECN: M-PCIe

Bit Location	Register Description	Attributes
14:12	**Downstream Component Presence** – This field indicates the presence and DRS status for the Downstream Component, if any, connected to the Link; defined values are: 000b Link Down – Presence Not Determined 001b Link Down – Component Not Present indicates the Downstream Port (DP) has determined that a Downstream Component is not present 010b Link Down – Component Present indicates the DP has determined that a Downstream Component is present, but the Data Link Layer is not active 011b Reserved 100b Link Up – Component Present indicates the DP has determined that a Downstream Component is present, but no DRS Message has been received since the Data Link Layer became active 101b Link Up – Component Present and DRS Received indicates the DP has received a DRS Message since the Data Link Layer became active 110b Reserved 111b Reserved Component Presence state must be determined by the logical "OR" of the Physical Layer in-band presence detect mechanism and, if present, any out-of-band presence detect mechanism implemented for the Link. If no out-of-band presence detect mechanism is implemented, then Component Presence state must be determined solely by the Physical Layer in-band presence detect mechanism. This field must be implemented in any Downstream Port where the DRS Supported bit is Set in the Link Capabilities 2 register. This field is RsvdZ for all other Functions. Default value of this field is 000b.	RO/RsvdZ
15	**DRS Message Received** – This bit must be Set whenever the Port receives a DRS Message. This bit must be Cleared in DL_Down. This bit must be implemented in any Downstream Port where the DRS Supported bit is Set in the Link Capabilities 2 register. This bit is RsvdZ for all other Functions. Default value of this bit is 0b.	RW1C / RsvdZ

Comment [VS275]: Updated per RC_ECN_RN_27_Aug_2013

Comment [VS276]: Updated per RC_ECN_RN_27_Aug_2013

7.8.21. Slot Capabilities 2 Register (Offset 34h)

This section is a placeholder. There are no capabilities that require this register.

This register must be treated by software as RsvdP.

7.8.22. Slot Control 2 Register (Offset 38h)

This section is a placeholder. There are no capabilities that require this register.

This register must be treated by software as RsvdP.

7.8.23. Slot Status 2 Register (Offset 3Ah)

This section is a placeholder. There are no capabilities that require this register.

This register must be treated by software as RsvdZ.

7.9. PCI Express Extended Capabilities

PCI Express Extended Capability registers are located in Configuration Space at offsets 256 or greater as shown in Figure 7-30 or in the Root Complex Register Block (RCRB). These registers when located in the Configuration Space are accessible using only the PCI Express Extended Configuration Space flat memory-mapped access mechanism.

PCI Express Extended Capability structures are allocated using a linked list of optional or required PCI Express Extended Capabilities following a format resembling PCI Capability structures. The first DWORD of the Capability structure identifies the Capability and version and points to the next Capability as shown in Figure 7-30.

Each Capability structure must be DWORD aligned.

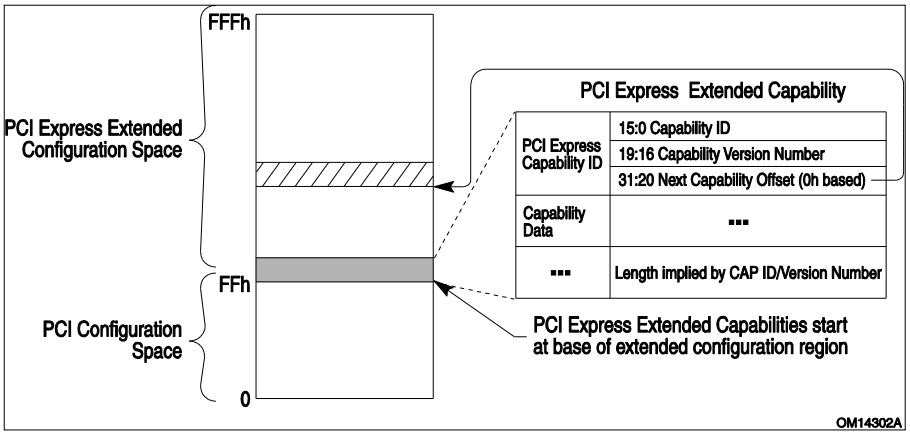


Figure 7-30: PCI Express Extended Configuration Space Layout

7.9.1. Extended Capabilities in Configuration Space

Extended Capabilities in Configuration Space always begin at offset 100h with a PCI Express Extended Capability header (Section 7.9.3). Absence of any Extended Capabilities is required to be indicated by an Extended Capability header with a Capability ID of 0000h, a Capability Version of 0h, and a Next Capability Offset of 000h.

7.9.2. Extended Capabilities in the Root Complex Register Block

Extended Capabilities in a Root Complex Register Block always begin at offset 000h with a PCI Express Extended Capability header (Section 7.9.3). Absence of any Extended Capabilities is required to be indicated by an Extended Capability header with a Capability ID of FFFFh and a Next Capability Offset of 000h.

7.9.3. PCI Express Extended Capability Header

All PCI Express Extended Capabilities must begin with a PCI Express Extended Capability header. Figure 7-31 details the allocation of register fields of a PCI Express Extended Capability header; Table 7-30 provides the respective bit definitions.

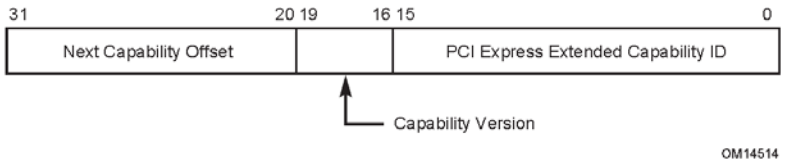


Figure 7-31: PCI Express Extended Capability Header

Table 7-30: PCI Express Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability.	RO

Bit Location	Register Description	Attributes
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. A version of the specification that changes the Extended Capability in a way that is not otherwise identifiable (e.g., through a new Capability field) is permitted to increment this field. All such changes to the Capability structure must be software-compatible. Software must check for Capability Version numbers that are greater than or equal to the highest number defined when the software is written, as Functions reporting any such Capability Version numbers will contain a Capability structure that is compatible with that piece of software.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits.	RO

7.10. Advanced Error Reporting Capability

The PCI Express Advanced Error Reporting Capability is an optional Extended Capability that may be implemented by PCI Express device Functions supporting advanced error control and reporting. The Advanced Error Reporting Capability structure definition has additional interpretation for Root Ports and Root Complex Event Collectors; software must interpret the Device/Port Type field in the PCI Express Capabilities register to determine the availability of additional registers for Root Ports and Root Complex Event Collectors.

Figure 7-32 shows the PCI Express Advanced Error Reporting Capability structure.

Note that if an error reporting bit field is marked as optional in the error registers, the bits must be implemented or not implemented as a group across the Status, Mask and Severity registers. In other words, a Function is required to implement the same error bit fields in corresponding Status, Mask and Severity registers. Bits corresponding to bit fields that are not implemented must be hardwired to 0, unless otherwise specified.

Except for Root Ports and Root Complex Event Collectors, if the End-End TLP Prefix Supported bit is Set, the Root Error Command and Error Source Identification registers must be RsvdP and the Root Error Status register must be RsvdZ.

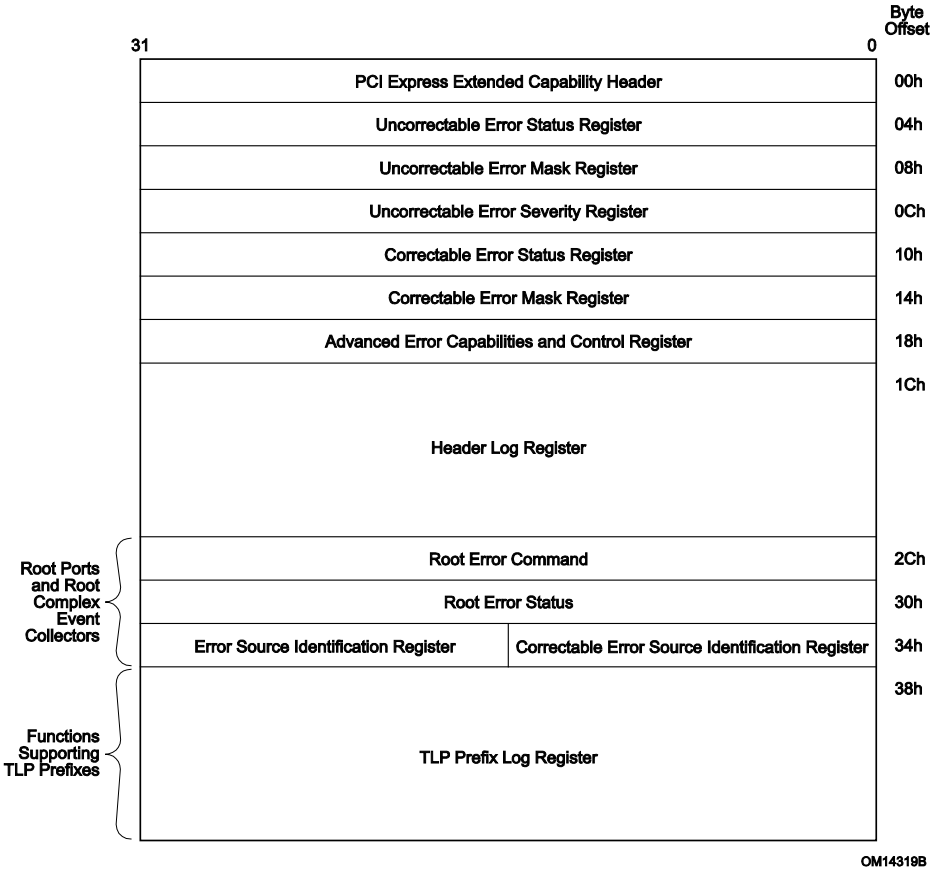


Figure 7-32: PCI Express Advanced Error Reporting Extended Capability Structure

7.10.1. Advanced Error Reporting Extended Capability Header (Offset 00h)

Figure 7-33 details the allocation of register fields of an Advanced Error Reporting Extended Capability header; Table 7-31 provides the respective bit definitions.

Refer to Section 7.9.3 for a description of the PCI Express Extended Capability header. The Extended Capability ID for the Advanced Error Reporting Capability is 0001h.

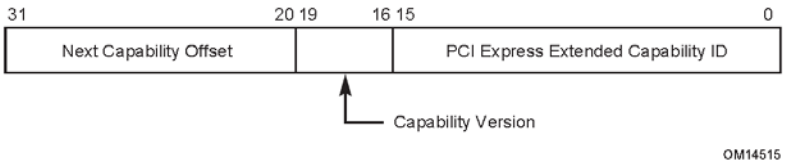


Figure 7-33: Advanced Error Reporting Extended Capability Header

Table 7-31: Advanced Error Reporting Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. The Extended Capability ID for the Advanced Error Reporting Capability is 0001h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. This field must be 2h if the End-End TLP Prefix Supported bit (see Section 7.8.15) is Set and must be 1h or 2h otherwise.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.10.2. Uncorrectable Error Status Register (Offset 04h)

The Uncorrectable Error Status register indicates error detection status of individual errors on a PCI Express device Function. An individual error status bit that is Set indicates that a particular error was detected; software may clear an error status by writing a 1b to the respective bit. Refer to Section 6.2 for further details. Register bits not implemented by the Function are hardwired to 0b. Figure 7-34 details the allocation of register fields of the Uncorrectable Error Status register; Table 7-32 provides the respective bit definitions.

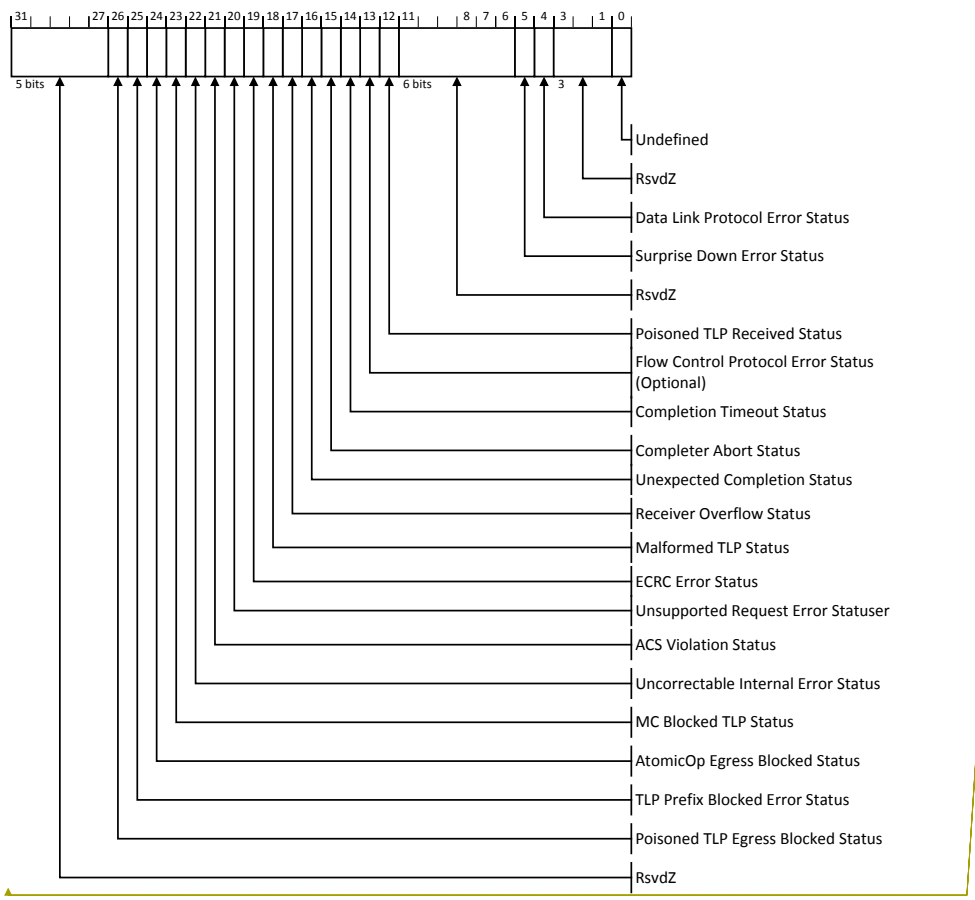


Figure 7-34: Uncorrectable Error Status Register

Table 7-32: Uncorrectable Error Status Register

Bit Location	Register Description	Attributes	Default
0	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate a Link Training Error. System software must ignore the value read from this bit. System software is permitted to write any value to this bit.	Undefined	Undefined
4	**Data Link Protocol Error Status**	RW1CS	0b
5	**Surprise Down Error Status** (Optional)	RW1CS	0b
12	**Poisoned TLP Received Status**	RW1CS	0b
13	**Flow Control Protocol Error Status** (Optional)	RW1CS	0b

Formatted: Font color: Black

Comment [VS277]: ECN: Enhanced DPC

Bit Location	Register Description	Attributes	Default
14	**Completion Timeout Status**[112]	RW1CS	0b
15	**Completer Abort Status** (Optional)	RW1CS	0b
16	**Unexpected Completion Status**	RW1CS	0b
17	**Receiver Overflow Status** (Optional)	RW1CS	0b
18	**Malformed TLP Status**	RW1CS	0b
19	**ECRC Error Status** (Optional)	RW1CS	0b
20	**Unsupported Request Error Status**	RW1CS	0b
21	**ACS Violation Status** (Optional)	RW1CS	0b
22	**Uncorrectable Internal Error Status** (Optional)	RW1CS	0b
23	**MC Blocked TLP Status** (Optional)	RW1CS	0b
24	**AtomicOp Egress Blocked Status** (Optional)	RW1CS	0b
25	**TLP Prefix Blocked Error Status** (Optional)	RW1CS	0b
26	**Poisoned TLP Egress Blocked Status** (Optional)	RW1CS	0b

Comment [VS278]: ECN: Enhanced DPC

7.10.3. Uncorrectable Error Mask Register (Offset 08h)

The Uncorrectable Error Mask register controls reporting of individual errors by the device Function to the PCI Express Root Complex via a PCI Express error Message. A masked error (respective bit Set in the mask register) is not recorded or reported in the Header Log ~~register~~, TLP Prefix Log , or~~does not update~~ ~~the~~ First Error Pointer, and is not reported to the PCI Express Root Complex by this Function. Refer to Section 6.2 for further details. There is a mask bit per error bit of the Uncorrectable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b. Figure 7-35 details the allocation of register fields of the Uncorrectable Error Mask register; Table 7-33 provides the respective bit definitions.

Comment [VS279]: B15

[112] For Switch Ports, required if the Switch Port issues Non-Posted Requests on its own behalf (vs. only forwarding such Requests generated by other devices). If the Switch Port does not issue such Requests, then the Completion Timeout mechanism is not applicable and this bit must be hardwired to 0b.

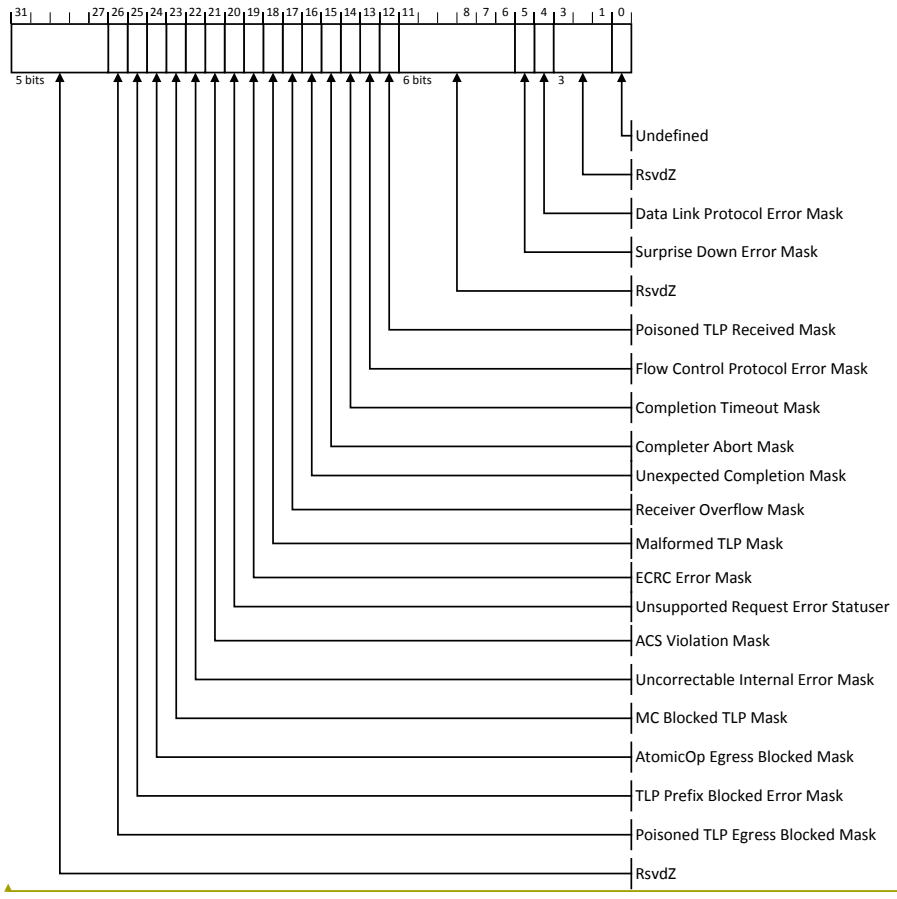


Figure 7-35: Uncorrectable Error Mask Register

Formatted: Font color: Black

Comment [VS280]: ECN: Enhanced DPC

Table 7-33: Uncorrectable Error Mask Register

Bit Location	Register Description	Attributes	Default
0	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to mask a Link Training Error. System software must ignore the value read from this bit. System software must only write a value of 1b to this bit.	Undefined	Undefined
4	**Data Link Protocol Error Mask**	RWS	0b
5	**Surprise Down Error Mask** (Optional)	RWS	0b
12	**Poisoned TLP Received Mask**	RWS	0b
13	**Flow Control Protocol Error Mask** (Optional)	RWS	0b
14	**Completion Timeout Mask**[113]	RWS	0b
15	**Completer Abort Mask** (Optional)	RWS	0b
16	**Unexpected Completion Mask**	RWS	0b
17	**Receiver Overflow Mask** (Optional)	RWS	0b
18	**Malformed TLP Mask**	RWS	0b
19	**ECRC Error Mask** (Optional)	RWS	0b
20	**Unsupported Request Error Mask**	RWS	0b
21	**ACS Violation Mask** (Optional)	RWS	0b
22	**Uncorrectable Internal Error Mask** (Optional)	RWS	1b
23	**MC Blocked TLP Mask** (Optional)	RWS	0b
24	**AtomicOp Egress Blocked Mask** (Optional)	RWS	0b
25	**TLP Prefix Blocked Error Mask** (Optional)	RWS	0b
26	**Poisoned TLP Egress Blocked Mask** (Optional)	RWS	1b

Comment [VS281]: ECN: Enhanced DPC

Comment [VS282]: ECN: Enhanced DPC

7.10.4. Uncorrectable Error Severity Register (Offset 0Ch)

The Uncorrectable Error Severity register controls whether an individual error is reported as a Non-fatal or Fatal error. An error is reported as fatal when the corresponding error bit in the severity register is Set. If the bit is Clear, the corresponding error is considered non-fatal. Refer to Section 6.2 for further details. Register fields for bits not implemented by the Function are hardwired to an implementation specific value. Figure 7-36 details the allocation of register fields of the Uncorrectable Error Severity register; Table 7-34 provides the respective bit definitions.

[113] For Switch Ports, required if the Switch Port issues Non-Posted Requests on its own behalf (vs. only forwarding such Requests generated by other devices). If the Switch Port does not issue such Requests, then the Completion Timeout mechanism is not applicable and this bit must be hardwired to 0b.

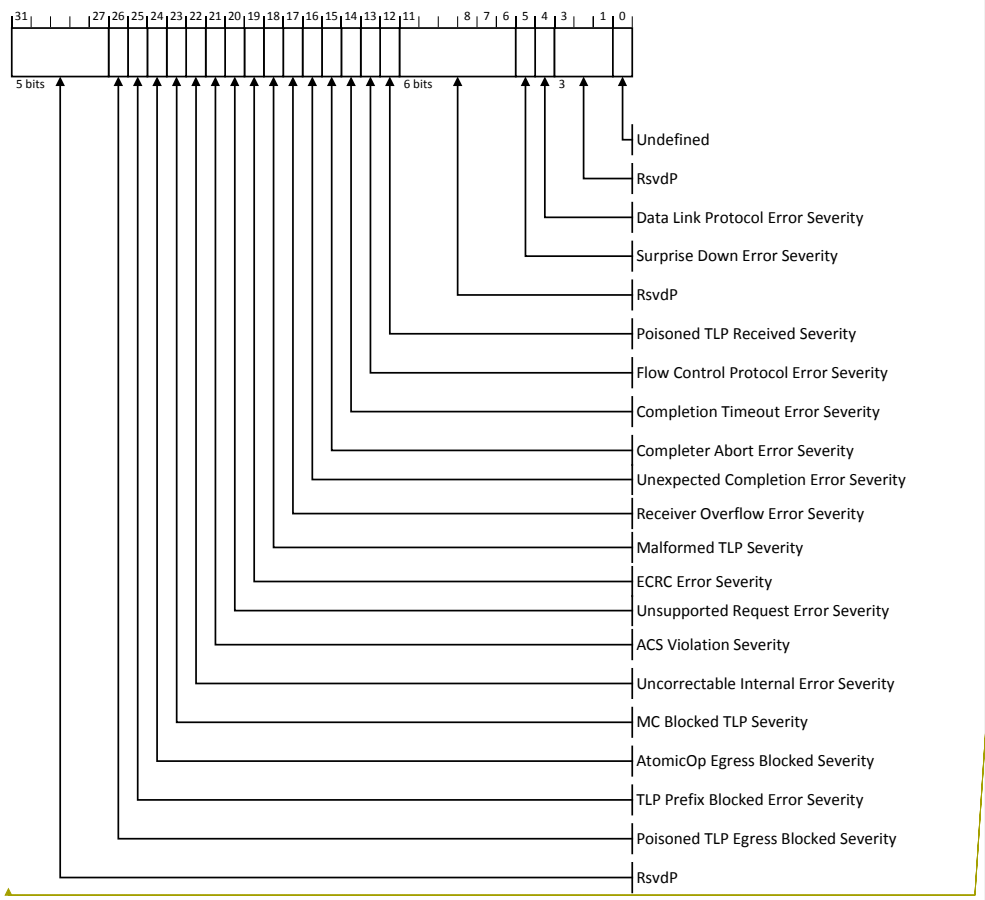


Figure 7-36: Uncorrectable Error Severity Register

Table 7-34: Uncorrectable Error Severity Register

Bit Location	Register Description	Attributes	Default
0	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to Set the severity of a Link Training Error. System software must ignore the value read from this bit. System software is permitted to write any value to this bit.	Undefined	Undefined
4	**Data Link Protocol Error Severity**	RWS	1b
5	**Surprise Down Error Severity** (Optional)	RWS	1b
12	**Poisoned TLP Received Severity**	RWS	0b
13	**Flow Control Protocol Error Severity** (Optional)	RWS	1b

Bit Location	Register Description	Attributes	Default
14	**Completion Timeout Error Severity**[114]	RWS	0b
15	**Completer Abort Error Severity** (Optional)	RWS	0b
16	**Unexpected Completion Error Severity**	RWS	0b
17	**Receiver Overflow Error Severity** (Optional)	RWS	1b
18	**Malformed TLP Severity**	RWS	1b
19	**ECRC Error Severity** (Optional)	RWS	0b
20	**Unsupported Request Error Severity**	RWS	0b
21	**ACS Violation Severity** (Optional)	RWS	0b
22	**Uncorrectable Internal Error Severity** (Optional)	RWS	1b
23	**MC Blocked TLP Severity** (Optional)	RWS	0b
24	**AtomicOp Egress Blocked Severity** (Optional)	RWS	0b
25	**TLP Prefix Blocked Error Severity** (Optional)	RWS	0b
26	**Poisoned TLP Egress Blocked Severity** (Optional)	RWS	0b

Comment [VS283]: ECN: Enhanced DPC

7.10.5. Correctable Error Status Register (Offset 10h)

The Correctable Error Status register reports error status of individual correctable error sources on a PCI Express device Function. When an individual error status bit is Set, it indicates that a particular error occurred; software may clear an error status by writing a 1b to the respective bit. Refer to Section 6.2 for further details. Register bits not implemented by the Function are hardwired to 0b. Figure 7-37 details the allocation of register fields of the Correctable Error Status register; Table 7-35 provides the respective bit definitions.

[114] For Switch Ports, required if the Switch Port issues Non-Posted Requests on its own behalf (vs. only forwarding such Requests generated by other devices). If the Switch Port does not issue such Requests, then the Completion Timeout mechanism is not applicable and this bit must be hardwired to 0b.

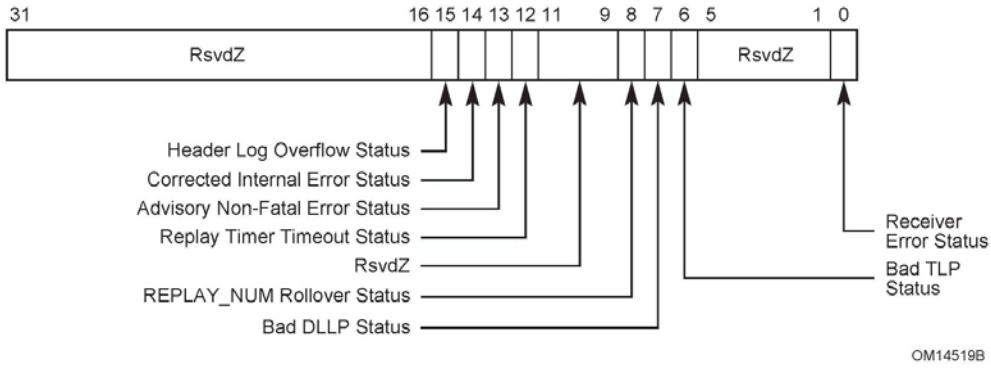


Figure 7-37: Correctable Error Status Register

Table 7-35: Correctable Error Status Register

Bit Location	Register Description	Attributes	Default
0	**Receiver Error Status**[115]	RW1CS	0b
6	**Bad TLP Status**	RW1CS	0b
7	**Bad DLLP Status**	RW1CS	0b
8	**REPLAY_NUM Rollover Status**	RW1CS	0b
12	**Replay Timer Timeout Status**	RW1CS	0b
13	**Advisory Non-Fatal Error Status**	RW1CS	0b
14	**Corrected Internal Error Status** (Optional)	RW1CS	0b
15	**Header Log Overflow Status** (Optional)	RW1CS	0b

7.10.6. Correctable Error Mask Register (Offset 14h)

The Correctable Error Mask register controls reporting of individual correctable errors by this Function to the PCI Express Root Complex via a PCI Express error Message. A masked error (respective bit Set in the mask register) is not reported to the PCI Express Root Complex by this Function. Refer to Section 6.2 for further details. There is a mask bit per error bit in the Correctable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b. Figure 7-38 details the allocation of register fields of the Correctable Error Mask register; Table 7-36 provides the respective bit definitions.

[115] For historical reasons, implementation of this bit is optional. If not implemented, this bit must be RsvdZ, and bit 0 of the Correctable Error Mask register must also not be implemented. Note that some checking for Receiver Errors is required in all cases (see Sections 4.2.1.1.3, 4.2.4.7, and 4.2.6).

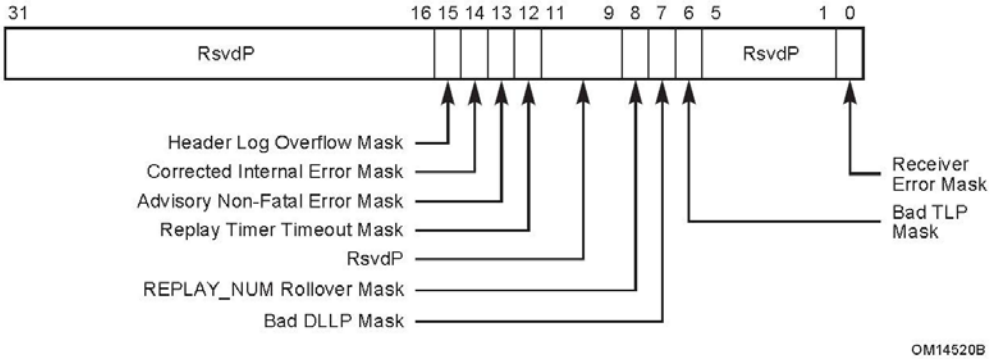


Figure 7-38: Correctable Error Mask Register

Table 7-36: Correctable Error Mask Register

Bit Location	Register Description	Attributes	Default
0	**Receiver Error Mask**[116]	RWS	0b
6	**Bad TLP Mask**	RWS	0b
7	**Bad DLLP Mask**	RWS	0b
8	**REPLAY_NUM Rollover Mask**	RWS	0b
12	**Replay Timer Timeout Mask**	RWS	0b
13	**Advisory Non-Fatal Error Mask** – This bit is Set by default to enable compatibility with software that does not comprehend Role-Based Error Reporting.	RWS	1b
14	**Corrected Internal Error Mask** (Optional)	RWS	1b
15	**Header Log Overflow Mask** (Optional)	RWS	1b

7.10.7. Advanced Error Capabilities and Control Register (Offset 18h)

Figure 7-39 details allocation of register fields in the Advanced Error Capabilities and Control register; Table 7-37 provides the respective bit definitions. Handling of multiple errors is discussed in Section 6.2.4.2.

[116] For historical reasons, implementation of this bit is optional. If not implemented, this bit must be RsvdP, and bit 0 of the Correctable Error Status register must also not be implemented. Note that some checking for Receiver Errors is required in all cases (see Sections 4.2.1.1.3, 4.2.4.7, and 4.2.6).

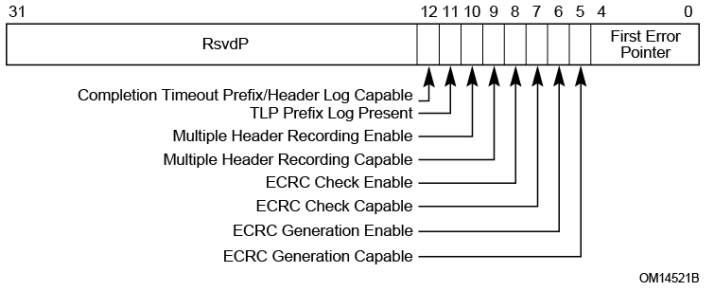


Figure 7-39: Advanced Error Capabilities and Control Register

Table 7-37: Advanced Error Capabilities and Control Register

Bit Location	Register Description	Attributes
4:0	**First Error Pointer** – The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register. Refer to Section 6.2 for further details.	ROS
5	**ECRC Generation Capable** – If Set, this bit indicates that the Function is capable of generating ECRC (see Section 2.7).	RO
6	**ECRC Generation Enable** – When Set, ECRC generation is enabled (see Section 2.7). Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RWS
7	**ECRC Check Capable** – If Set, this bit indicates that the Function is capable of checking ECRC (see Section 2.7).	RO
8	**ECRC Check Enable** – When Set, ECRC checking is enabled (see Section 2.7). Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RWS
9	**Multiple Header Recording Capable** – If Set, this bit indicates that the Function is capable of recording more than one error header. Refer to Section 6.2 for further details.	RO
10	**Multiple Header Recording Enable** – When Set, this bit enables the Function to record more than one error header. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RWS

Bit Location	Register Description	Attributes
11	**TLP Prefix Log Present** – If Set and the First Error Pointer is valid, indicates that the TLP Prefix Log register contains valid information. If Clear or if First Error Pointer is invalid, the TLP Prefix Log register is undefined. Default value of this bit is 0. This bit is RsvdP if the End-End TLP Prefix Supported bit is Clear.	ROS
12	**Completion Timeout Prefix/Header Log Capable** – If Set, this bit indicates that the Function records the prefix/header of Request TLPs that experience a Completion Timeout error.	RO

Comment [VS284]: ECN: Downstream Port containment (DPC)

7.10.8. Header Log Register (Offset 1Ch)

The Header Log register contains the header for the TLP corresponding to a detected error; refer to Section 6.2 for further details. Section 6.2 also describes the conditions where the packet header is recorded. This register is 16 bytes and adheres to the format of the headers defined throughout this specification.

The header is captured such that, when read using DW accesses, the fields of the header are laid out in the same way the headers are presented in this document. Therefore, byte 0 of the header is located in byte 3 of the Header Log register, byte 1 of the header is in byte 2 of the Header Log register and so forth. For 12-byte headers, only bytes 0 through 11 of the Header Log register are used and values in bytes 12 through 15 are undefined.

In certain cases where a Malformed TLP is reported, the Header Log register may contain TLP Prefix information. See Section 6.2.4.4 for details.

Figure 7-40 details allocation of register fields in the Header Log register; Table 7-38 provides the respective bit definitions.

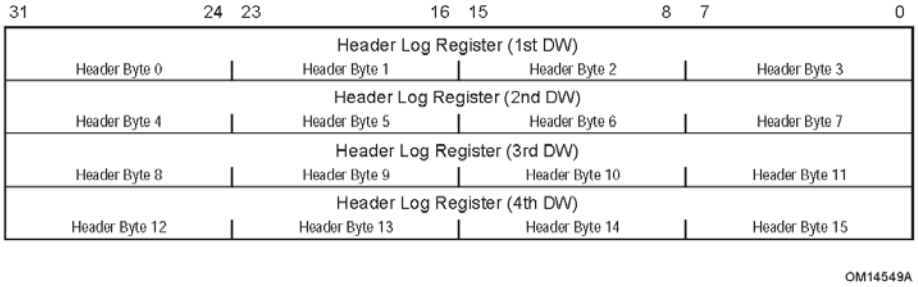


Figure 7-40: Header Log Register

Table 7-38: Header Log Register

Bit Location	Register Description	Attributes	Default
127:0	Header of TLP associated with error	ROS	0

7.10.9. Root Error Command Register (Offset 2Ch)

The Root Error Command register allows further control of Root Complex response to Correctable, Non-Fatal, and Fatal error Messages than the basic Root Complex capability to generate system errors in response to error Messages (either received or internally generated). Bit fields (see Figure 7-41) enable or disable generation of interrupts (claimed by the Root Port or Root Complex Event Collector) in addition to system error Messages according to the definitions in Table 7-39.

For both Root Ports and Root Complex Event Collectors, in order for a received error Message or an internally generated error Message to generate an interrupt enabled by this register, the error Message must be enabled for "transmission" by the Root Port or Root Complex Event Collector (see Section 6.2.4.1 and Section 6.2.8.1).

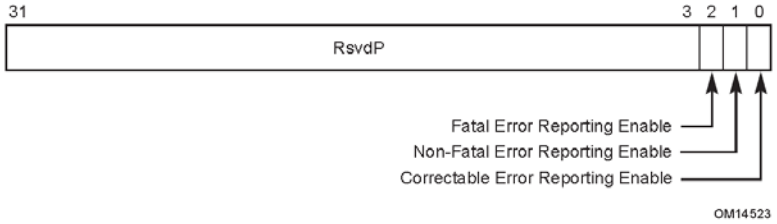


Figure 7-41: Root Error Command Register

Table 7-39: Root Error Command Register

Bit Location	Register Description	Attributes	Default
0	**Correctable Error Reporting Enable** – When Set, this bit enables the generation of an interrupt when a correctable error is reported by any of the Functions in the ~~hierarchy~~ Hierarchy Domain associated with this Root Port. Root Complex Event Collectors provide support for the above described functionality for Root Complex Integrated Endpoints. Refer to Section 6.2 for further details.	RW	0b
1	**Non-Fatal Error Reporting Enable** – When Set, this bit enables the generation of an interrupt when a Non-fatal error is reported by any of the Functions in the ~~hierarchy~~ Hierarchy Domain associated with this Root Port. Root Complex Event Collectors provide support for the above described functionality for Root Complex Integrated Endpoints. Refer to Section 6.2 for further details.	RW	0b
2	**Fatal Error Reporting Enable** – When Set, this bit enables the generation of an interrupt when a Fatal error is reported by any of the Functions in the ~~hierarchy~~ Hierarchy Domain associated with this Root Port. Root Complex Event Collectors provide support for the above described functionality for Root Complex Integrated Endpoints. Refer to Section 6.2 for further details.	RW	0b

Comment [VS285]: A20

Comment [VS286]: A20

Comment [VS287]: A20

System error generation in response to PCI Express error Messages may be turned off by system software using the PCI Express Capability structure described in Section 7.8 when advanced error reporting via interrupts is enabled. Refer to Section 6.2 for further details.

7.10.10. Root Error Status Register (Offset 30h)

The Root Error Status register reports status of error Messages (ERR_COR, ERR_NONFATAL, and ERR_FATAL) received by the Root Port, and of errors detected by the Root Port itself (which are treated conceptually as if the Root Port had sent an error Message to itself). In order to update this register, error Messages received by the Root Port and/or internally generated error Messages must be enabled for "transmission" by the primary interface of the Root Port. ERR_NONFATAL and ERR_FATAL Messages are grouped together as uncorrectable. Each correctable and uncorrectable (Non-fatal and Fatal) error source has a first error bit and a next error bit associated with it respectively. When an error is received by a Root Complex, the respective first error bit is Set and the Requester ID is logged in the Error Source Identification register. A Set individual error status bit indicates that a particular error category occurred; software may clear an error status by writing a 1b to the respective bit. If software does not clear the first reported error before another

error Message is received of the same category (correctable or uncorrectable), the corresponding next error status bit will be set but the Requester ID of the subsequent error Message is discarded. The next error status bits may be cleared by software by writing a 1b to the respective bit as well. Refer to Section 6.2 for further details. This register is updated regardless of the settings of the Root Control register and the Root Error Command register. Figure 7-42 details allocation of register fields in the Root Error Status register; Table 7-40 provides the respective bit definitions. Root Complex Event Collectors provide support for the above-described functionality for Root Complex Integrated Endpoints (and for the Root Complex Event Collector itself). In order to update this register, error Messages received by the Root Complex Event Collector from its associated Root Complex Integrated Endpoints and/or internally generated error Messages must be enabled for "transmission" by the Root Complex Event Collector.

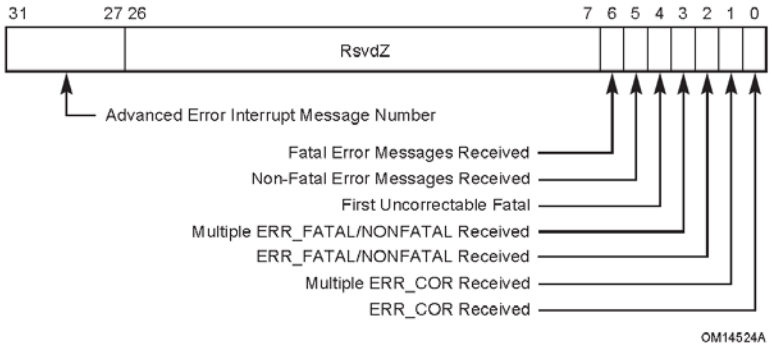


Figure 7-42: Root Error Status Register

Table 7-40: Root Error Status Register

Bit Location	Register Description	Attributes
0	**ERR_COR Received** – Set when a Correctable error Message is received and this bit is not already Set. Default value of this bit is 0b.	RW1CS
1	**Multiple ERR_COR Received** – Set when a Correctable error Message is received and ERR_COR Received is already Set. Default value of this bit is 0b.	RW1CS
2	**ERR_FATAL/NONFATAL Received** – Set when either a Fatal or a Non-fatal error Message is received and this bit is not already Set. Default value of this bit is 0b.	RW1CS
3	**Multiple ERR_FATAL/NONFATAL Received** – Set when either a Fatal or a Non-fatal error is received and **ERR_FATAL/NONFATAL Received** is already Set. Default value of this bit is 0b.	RW1CS

Bit Location	Register Description	Attributes
4	**First Uncorrectable Fatal** – Set when the first Uncorrectable error Message received is for a Fatal error. Default value of this field is 0b.	RW1CS
5	**Non-Fatal Error Messages Received** – Set when one or more Non-Fatal Uncorrectable error Messages have been received. Default value of this bit is 0b.	RW1CS
6	**Fatal Error Messages Received** – Set when one or more Fatal Uncorrectable error Messages have been received. Default value of this bit is 0b.	RW1CS
31:27	**Advanced Error Interrupt Message Number** – This register indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this Capability. For MSI, the value in this register indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the MSI Message Control register. For MSI-X, the value in this register indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this register must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this register must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this register is undefined.	RO

7.10.11. Error Source Identification Register (Offset 34h)

The Error Source Identification register identifies the source (Requester ID) of first correctable and uncorrectable (Non-fatal/Fatal) errors reported in the Root Error Status register. Refer to Section 6.2 for further details. This register is updated regardless of the settings of the Root Control register and the Root Error Command register. Figure 7-43 details allocation of register fields in the Error Source Identification register; Table 7-41 provides the respective bit definitions.

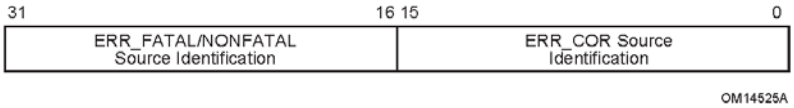


Figure 7-43: Error Source Identification Register

Table 7-41: Error Source Identification Register

Bit Location	Register Description	Attributes
15:0	**ERR_COR Source Identification** – Loaded with the Requester ID indicated in the received ERR_COR Message when the ERR_COR Received bit is not already set. Default value of this field is 0000h.	ROS
31:16	**ERR_FATAL/NONFATAL Source Identification** – Loaded with the Requester ID indicated in the received ERR_FATAL or ERR_NONFATAL Message when the ERR_FATAL/NONFATAL Received bit is not already set. Default value of this field is 0000h.	ROS

7.10.12. TLP Prefix Log Register (Offset 38h)

The TLP Prefix Log register captures the End-End TLP Prefix(s) for the TLP corresponding to the detected error; refer to Section 6.2 for further details. The TLP Prefix Log register is only meaningful when the TLP Prefix Log Present bit is Set (see Section 7.10.7).

The TLP Prefixes are captured such that, when read using DW accesses, the fields of the TLP Prefix are laid out in the same way the fields of the TLP Prefix are described. Therefore, byte 0 of a TLP Prefix is located in byte 3 of the associated TLP Prefix Log register; byte 1 of a TLP Prefix is located in byte 2; and so forth.

The First TLP Prefix Log register contains the first End-End TLP Prefix from the TLP (see Section 6.2.4.4). The Second TLP Prefix Log register contains the second End-End TLP Prefix and so forth. If the TLP contains fewer than four End-End TLP Prefixes, the remaining TLP Prefix Log registers contain zero. A TLP that contains more End-End TLP Prefixes than are indicated by the Function's Max End-End TLP Prefixes field must be handled as an error (see Section 2.2.10.2 for specifics). To allow software to detect this condition, the supported number of End-End TLP

Prefixes are logged in this register, the first overflow End-End TLP Prefix is logged in the first DW of the Header Log register and the remaining DWs of the Header Log register are undefined (see Section 6.2.4.4).

The TLP Prefix Log registers beyond the number supported by the Function are hardwired to zero. For example, if a Functions, Max End-End TLP Prefixes field contains 10b (indicating 2 DW of buffering) then the third and fourth TLP Prefix Log registers are hardwired to zero. If the End-End TLP Prefix Supported bit (Section 7.8.15) is Clear, the TLP Prefix Log register is not required to be implemented.

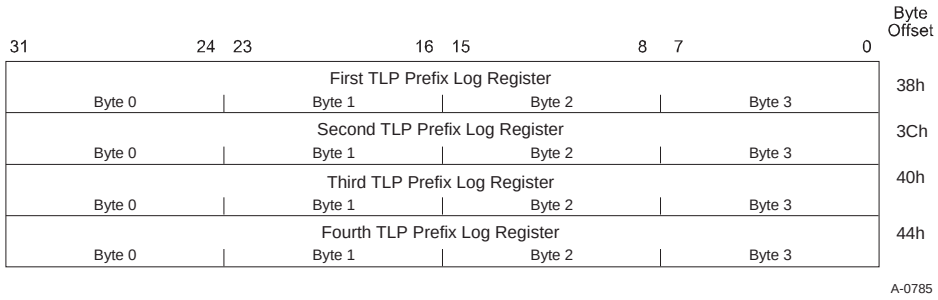


Figure 7-44: TLP Prefix Log Register

Table 7-42: TLP Prefix Log Register

Bit Location	Register Description	Attributes	Default
127:0	TLP Prefix Log	ROS	0

7.11. Virtual Channel Capability

The Virtual Channel (VC) Capability is an optional Extended Capability required for devices that have Ports (or for individual Functions) that support functionality beyond the default Traffic Class (TC0) over the default Virtual Channel (VC0). This may apply to devices with only one VC that support TC filtering or to devices that support multiple VCs. Note that a PCI Express device that supports only TC0 over VC0 does not require VC Extended Capability and associated registers. Figure 7-45 provides a high level view of the PCI Express Virtual Channel Capability structure. This structure controls Virtual Channel assignment for PCI Express Links and may be present in any device (or RCRB) that contains (controls) a Port, or any device that has a Multi-Function Virtual Channel (MFVC) Capability structure. Some registers/fields in the PCI Express Virtual Channel Capability structure may have different interpretation for Endpoints, Switch Ports, Root Ports and RCRB. Software must interpret the Device/Port Type field in the PCI Express Capabilities register to determine the availability and meaning of these registers/fields.

The PCI Express VC Capability structure is permitted in the Extended Configuration Space of all single-Function devices or in RCRBs.

A multi-Function device at an Upstream Port is permitted to optionally contain a Multi-Function Virtual Channel (MFVC) Capability structure (see Section 7.18). If a multi-Function device contains

an MFVC Capability structure, any or all of its Functions are permitted to contain a VC Capability structure. Per-Function VC Capability structures are also permitted for devices inside a Switch that contain only Switch Downstream Port Functions, or for Root Complex Integrated Endpoints. Otherwise, only Function 0 is permitted to contain a VC Capability structure.

To preserve software backward compatibility, two Extended Capability IDs are permitted for VC Capability structures: 0002h and 0009h. Any VC Capability structure in a device that also contains an MFVC Capability structure must use the Extended Capability ID 0009h. A VC Capability structure in a device that does not contain an MFVC Capability structure must use the Extended Capability ID 0002h.

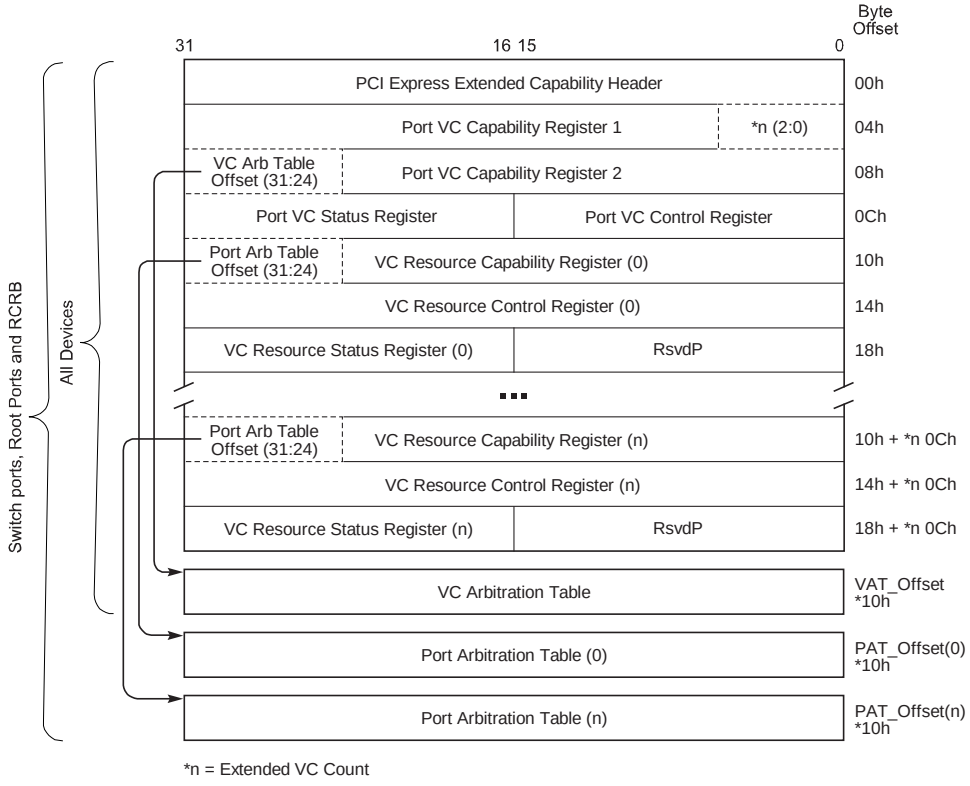


Figure 7-45: PCI Express Virtual Channel Capability Structure

The following sections describe the registers/fields of the PCI Express Virtual Channel Capability structure.

7.11.1. Virtual Channel Extended Capability Header (Offset 00h)

Refer to Section 7.9.3 for a description of the PCI Express Extended Capability header. A Virtual Channel Capability must use one of two Extended Capability IDs: 0002h or 0009h. Refer to Section 7.11 for rules governing when each should be used. Figure 7-46 details allocation of register fields in the Virtual Channel Extended Capability header; Table 7-43 provides the respective bit definitions.

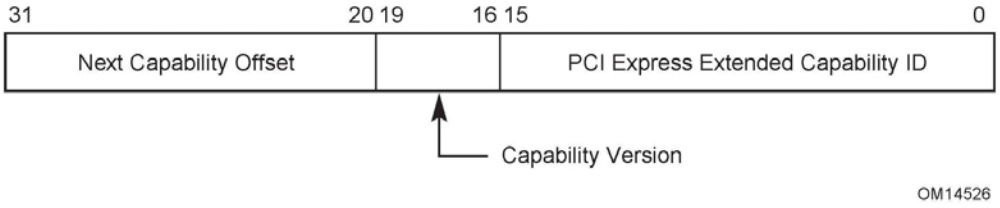


Figure 7-46: Virtual Channel Extended Capability Header

Table 7-437-43: Virtual Channel Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Virtual Channel Capability is either 0002h or 0009h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.11.2. Port VC Capability Register 1 (Offset 04h)

The Port VC Capability register 1 describes the configuration of the Virtual Channels associated with a PCI Express Port. Figure 7-47 details allocation of register fields in the Port VC Capability register 1; Table 7-44 provides the respective bit definitions.

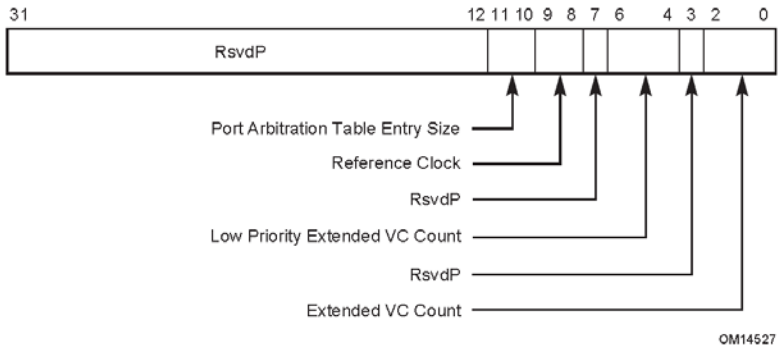


Figure 7-47: Port VC Capability Register 1

Table 7-44: Port VC Capability Register 1

Bit Location	Register Description	Attributes
2:0	**Extended VC Count** – Indicates the number of (extended) Virtual Channels in addition to the default VC supported by the device. This field is valid for all Functions. The minimum value of this field is 000b (for devices that only support the default VC). The maximum value is 7.	RO
6:4	**Low Priority Extended VC Count** – Indicates the number of (extended) Virtual Channels in addition to the default VC belonging to the low-priority VC (LPVC) group that has the lowest priority with respect to other VC resources in a strict-priority VC Arbitration. This field is valid for all Functions. The minimum value of this field is 000b and the maximum value is Extended VC Count.	RO
9:8	**Reference Clock** – Indicates the reference clock for Virtual Channels that support time-based WRR Port Arbitration. This field is valid for RCRBs, Switch Ports, and Root Ports that support peer-to-peer traffic. It is not valid for Root Ports that do not support peer-to-peer traffic, Endpoints, and Switches or Root Complexes not implementing WRR, and must be hardwired to 00b. Defined encodings are: 00b 100 ns reference clock 01b – 11b Reserved	RO

Bit Location	Register Description	Attributes
11:10	**Port Arbitration Table Entry Size** – Indicates the size (in bits) of Port Arbitration table entry in the Function. This field is valid only for RCRBs, Switch Ports, and Root Ports that support peer-to-peer traffic. It is not valid and must be hardwired to 00b for Root Ports that do not support peer-to-peer traffic and Endpoints. Defined encodings are: 00b The size of Port Arbitration table entry is 1 bit. 01b The size of Port Arbitration table entry is 2 bits. 10b The size of Port Arbitration table entry is 4 bits. 11b The size of Port Arbitration table entry is 8 bits.	RO

7.11.3. Port VC Capability Register 2 (Offset 08h)

The Port VC Capability register 2 provides further information about the configuration of the Virtual Channels associated with a PCI Express Port. Figure 7-48 details allocation of register fields in the Port VC Capability register 2; Table 7-45 provides the respective bit definitions.



Figure 7-48: Port VC Capability Register 2

Table 7-45: Port VC Capability Register 2

Bit Location	Register Description	Attributes
7:0	**VC Arbitration Capability** – Indicates the types of VC Arbitration supported by the Function for the LPVC group. This field is valid for all Functions that report a Low Priority Extended VC Count field greater than 0. For all other Functions, this field must be hardwired to 00h. Each bit location within this field corresponds to a VC Arbitration Capability defined below. When more than 1 bit in this field is Set, it indicates that the Port can be configured to provide different VC arbitration services. Defined bit positions are: Bit 0 – Hardware fixed arbitration scheme, e.g., Round Robin Bit 1 – Weighted Round Robin (WRR) arbitration with 32 phases Bit 2 – WRR arbitration with 64 phases Bit 3 – WRR arbitration with 128 phases Bits 4-7 – Reserved	RO
31:24	**VC Arbitration Table Offset** – Indicates the location of the VC Arbitration Table. This field is valid for all Functions. This field contains the zero-based offset of the table in DQWORDS (16 bytes) from the base address of the Virtual Channel Capability structure. A value of 0 indicates that the table is not present.	RO

7.11.4. Port VC Control Register (Offset 0Ch)

Figure 7-49 details allocation of register fields in the Port VC Control register; Table 7-46 provides the respective bit definitions.

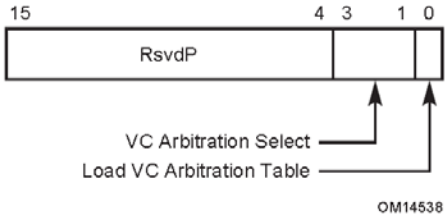


Figure 7-49: Port VC Control Register

Table 7-46: Port VC Control Register

Bit Location	Register Description	Attributes
0	**Load VC Arbitration Table** – Used by software to update the VC Arbitration Table. This bit is valid for all Functions when the selected VC Arbitration uses the VC Arbitration Table. Software sets this bit to request hardware to apply new values programmed into VC Arbitration Table; clearing this bit has no effect. Software checks the VC Arbitration Table Status bit to confirm that new values stored in the VC Arbitration Table are latched by the VC arbitration logic. This bit always returns 0b when read.	RW
3:1	**VC Arbitration Select** – Used by software to configure the VC arbitration by selecting one of the supported VC Arbitration schemes indicated by the VC Arbitration Capability field in the Port VC Capability register 2. This field is valid for all Functions. The permissible values of this field are numbers corresponding to one of the asserted bits in the VC Arbitration Capability field. This field cannot be modified when more than one VC in the LPVC group is enabled.	RW

7.11.5. Port VC Status Register (Offset 0Eh)

The Port VC Status register provides status of the configuration of Virtual Channels associated with a Port. Figure 7-50 details allocation of register fields in the Port VC Status register; Table 7-47 provides the respective bit definitions.

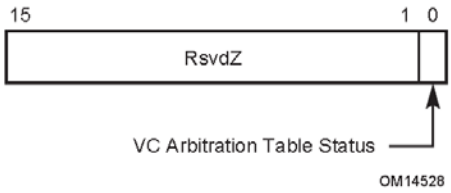


Figure 7-50: Port VC Status Register

Table 7-47: Port VC Status Register

Bit Location	Register Description	Attributes
0	**VC Arbitration Table Status** – Indicates the coherency status of the VC Arbitration Table. This bit is valid for all Functions when the selected VC uses the VC Arbitration Table. This bit is Set by hardware when any entry of the VC Arbitration Table is written by software. This bit is Cleared by hardware when hardware finishes loading values stored in the VC Arbitration Table after software sets the Load VC Arbitration Table bit in the Port VC Control register. Default value of this bit is 0b.	RO

7.11.6. VC Resource Capability Register

The VC Resource Capability register describes the capabilities and configuration of a particular Virtual Channel resource. Figure 7-51 details allocation of register fields in the VC Resource Capability register; Table 7-48 provides the respective bit definitions.

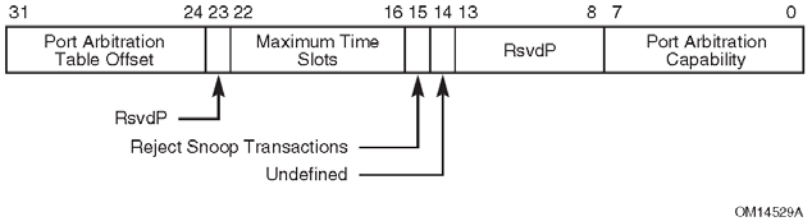


Figure 7-51: VC Resource Capability Register

Table 7-48: VC Resource Capability Register

Bit Location	Register Description	Attributes
7:0	**Port Arbitration Capability** – Indicates types of Port Arbitration supported by the VC resource. This field is valid for all Switch Ports, Root Ports that support peer-to-peer traffic, and RCRBs, but not for Endpoints or Root Ports that do not support peer-to-peer traffic. Each bit location within this field corresponds to a Port Arbitration Capability defined below. When more than 1 bit in this field is Set, it indicates that the VC resource can be configured to provide different arbitration services. Software selects among these capabilities by writing to the Port Arbitration Select field (see Section 7.11.7). Defined bit positions are: Bit 0 – Non-configurable hardware-fixed arbitration scheme, e.g., Round Robin (RR) Bit 1 – Weighted Round Robin (WRR) arbitration with 32 phases Bit 2 – WRR arbitration with 64 phases Bit 3 – WRR arbitration with 128 phases Bit 4 – Time-based WRR with 128 phases Bit 5 – WRR arbitration with 256 phases Bits 6-7 – Reserved	RO
14	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate Advanced Packet Switching. System software must ignore the value read from this bit.	RO
15	**Reject Snoop Transactions** – When Clear, transactions with or without the No Snoop bit Set within the TLP header are allowed on this VC. When Set, any transaction for which the No Snoop attribute is applicable but is not Set within the TLP header is permitted to be rejected as an Unsupported Request. Refer to Section 2.2.6.5 for information on where the No Snoop attribute is applicable. This bit is valid for Root Ports and RCRB; it is not valid for Endpoints or Switch Ports.	HwInit
22:16	**Maximum Time Slots** – Indicates the maximum number of time slots (minus one) that the VC resource is capable of supporting when it is configured for time-based WRR Port Arbitration. For example, a value 000 0000b in this field indicates the supported maximum number of time slots is 1 and a value of 111 1111b indicates the supported maximum number of time slots is 128. This field is valid for all Switch Ports, Root Ports that support peer-to-peer traffic, and RCRBs, but is not valid for Endpoints or Root Ports that do not support peer-to-peer traffic. In addition, this field is valid only when the Port Arbitration Capability field indicates that the VC resource supports time-based WRR Port Arbitration.	HwInit

Bit Location	Register Description	Attributes
31:24	**Port Arbitration Table Offset** – Indicates the location of the Port Arbitration Table associated with the VC resource. This field is valid for all Switch Ports, Root Ports that support peer-to-peer traffic, and RCRBs, but is not valid for Endpoints or Root Ports that do not support peer-to-peer traffic. This field contains the zero-based offset of the table in DQWORDS (16 bytes) from the base address of the Virtual Channel Capability structure. A value of 00h indicates that the table is not present.	RO

7.11.7. VC Resource Control Register

Figure 7-52 details allocation of register fields in the VC Resource Control register; Table 7-49 provides the respective bit definitions.

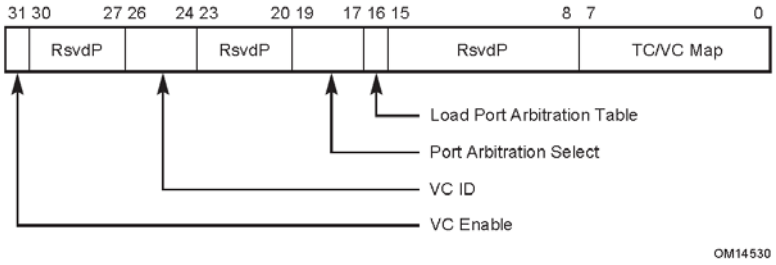


Figure 7-52: VC Resource Control Register

Table 7-49: VC Resource Control Register

Bit Location	Register Description	Attributes
7:0	**TC/VC Map** – This field indicates the TCs that are mapped to the VC resource. This field is valid for all Functions. Bit locations within this field correspond to TC values. For example, when bit 7 is Set in this field, TC7 is mapped to this VC resource. When more than 1 bit in this field is Set, it indicates that multiple TCs are mapped to the VC resource. In order to remove one or more TCs from the TC/VC Map of an enabled VC, software must ensure that no new or outstanding transactions with the TC labels are targeted at the given Link. Default value of this field is FFh for the first VC resource and is 00h for other VC resources. Note: Bit 0 of this field is read-only. It must be Set for the default VC0 and Clear for all other enabled VCs.	RW (see the note for exceptions)

Bit Location	Register Description	Attributes
16	**Load Port Arbitration Table** – When Set, this bit updates the Port Arbitration logic from the Port Arbitration Table for the VC resource. This bit is valid for all Switch Ports, Root Ports that support peer-to-peer traffic, and RCRBs, but is not valid for Endpoints or Root Ports that do not support peer-to-peer traffic. In addition, this bit is only valid when the Port Arbitration Table is used by the selected Port Arbitration scheme (that is indicated by a Set bit in the Port Arbitration Capability field selected by Port Arbitration Select). Software sets this bit to signal hardware to update Port Arbitration logic with new values stored in Port Arbitration Table; clearing this bit has no effect. Software uses the Port Arbitration Table Status bit to confirm whether the new values of Port Arbitration Table are completely latched by the arbitration logic. This bit always returns 0b when read. Default value of this bit is 0b.	RW
19:17	**Port Arbitration Select** – This field configures the VC resource to provide a particular Port Arbitration service. This field is valid for RCRBs, Root Ports that support peer-to-peer traffic, and Switch Ports, but is not valid for Endpoints or Root Ports that do not support peer-to-peer traffic. The permissible value of this field is a number corresponding to one of the asserted bits in the Port Arbitration Capability field of the VC resource.	RW
26:24	**VC ID** – This field assigns a VC ID to the VC resource (see note for exceptions). This field is valid for all Functions. This field cannot be modified when the VC is already enabled. Note: For the first VC resource (default VC), this field is read-only and must be hardwired to 000b.	RW

Bit Location	Register Description	Attributes
31	**VC Enable** – This bit, when Set, enables a Virtual Channel (see note 1 for exceptions). The Virtual Channel is disabled when this bit is cleared. This bit is valid for all Functions. Software must use the VC Negotiation Pending bit to check whether the VC negotiation is complete. Default value of this bit is 1b for the first VC resource and is 0b for other VC resource(s). Notes: 1. This bit is hardwired to 1b for the default VC (VC0), i.e., writing to this bit has no effect for VC0. 2. To enable a Virtual Channel, the VC Enable bits for that Virtual Channel must be Set in both components on a Link. 3. To disable a Virtual Channel, the VC Enable bits for that Virtual Channel must be cleared in both components on a Link. 4. Software must ensure that no traffic is using a Virtual Channel at the time it is disabled. 5. Software must fully disable a Virtual Channel in both components on a Link before re-enabling the Virtual Channel.	RW

7.11.8. VC Resource Status Register

Figure 7-53 details allocation of register fields in the VC Resource Status register; Table 7-50 provides the respective bit definitions.

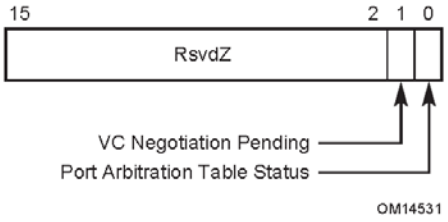


Figure 7-53: VC Resource Status Register

Table 7-50: VC Resource Status Register

Bit Location	Register Description	Attributes
0	**Port Arbitration Table Status** – This bit indicates the coherency status of the Port Arbitration Table associated with the VC resource. This bit is valid for RCRBs, Root Ports that support peer-to-peer traffic, and Switch Ports, but is not valid for Endpoints or Root Ports that do not support peer-to-peer traffic. In addition, this bit is valid only when the Port Arbitration Table is used by the selected Port Arbitration for the VC resource. This bit is Set by hardware when any entry of the Port Arbitration Table is written to by software. This bit is Cleared by hardware when hardware finishes loading values stored in the Port Arbitration Table after software sets the Load Port Arbitration Table bit. Default value of this bit is 0b.	RO
1	**VC Negotiation Pending** –This bit indicates whether the Virtual Channel negotiation (initialization or disabling) is in pending state. This bit is valid for all Functions. The value of this bit is defined only when the Link is in the DL_Active state and the Virtual Channel is enabled (its VC Enable bit is Set). When this bit is Set by hardware, it indicates that the VC resource has not completed the process of negotiation. This bit is Cleared by hardware after the VC negotiation is complete (on exit from the FC_INIT2 state). For VC0, this bit is permitted to be hardwired to 0b. Before using a Virtual Channel, software must check whether the VC Negotiation Pending bits for that Virtual Channel are Clear in both components on the Link.	RO

7.11.9. VC Arbitration Table

The VC Arbitration Table is a read-write register array that is used to store the arbitration table for VC Arbitration. This register array is valid for all Functions when the selected VC Arbitration uses a WRR table. If it exists, the VC Arbitration Table is located by the VC Arbitration Table Offset field.

The VC Arbitration Table is a register array with fixed-size entries of 4 bits. Figure 7-54 depicts the table structure of an example VC Arbitration Table with 32 phases. Each 4-bit table entry corresponds to a phase within a WRR arbitration period. The definition of table entry is depicted in Table 7-51. The lower 3 bits (bits 0-2) contain the VC ID value, indicating that the corresponding phase within the WRR arbitration period is assigned to the Virtual Channel indicated by the VC ID (must be a valid VC ID that corresponds to an enabled VC).

The highest bit (bit 3) of the table entry is Reserved. The length of the table depends on the selected VC Arbitration as shown in Table 7-52.

When the VC Arbitration Table is used by the default VC Arbitration method, the default values of the table entries must be all zero to ensure forward progress for the default VC (with VC ID of 0).

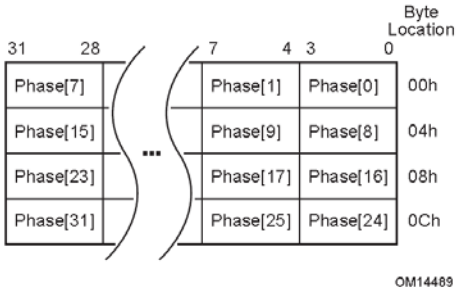


Figure 7-54: Example VC Arbitration Table with 32 Phases

Table 7-51: Definition of the 4-bit Entries in the VC Arbitration Table

Bit Location	Description	Attributes
2:0	VC ID	RW
3	RsvdP	RW

Table 7-52: Length of the VC Arbitration Table

VC Arbitration Select	VC Arbitration Table Length (in # of Entries)
001b	32
010b	64
011b	128

7.11.10. Port Arbitration Table

The Port Arbitration Table register is a read-write register array that is used to store the WRR or time-based WRR arbitration table for Port Arbitration for the VC resource. This register array is valid for all Switch Ports, Root Ports that support peer-to-peer traffic, and RCRBs, but is not valid for Endpoints or Root Ports that do not support peer-to-peer traffic. It is only present when one or more asserted bits in the Port Arbitration Capability field indicate that the component supports a Port Arbitration scheme that uses a programmable arbitration table. Furthermore, it is only valid when one of the above-mentioned bits in the Port Arbitration Capability field is selected by the Port Arbitration Select field.

The Port Arbitration Table represents one Port arbitration period. Figure 7-55 shows the structure of an example Port Arbitration Table with 128 phases and 2-bit table entries. Each table entry containing a Port Number corresponds to a phase within a Port arbitration period. For example, a table with 2-bit entries can be used by a Switch component with up to four Ports. A Port Number written to a table entry indicates that the phase within the Port Arbitration period is assigned to the selected PCI Express Port (the Port Number must be a valid one).

- When the WRR Port Arbitration is used for a VC of any Egress Port, at each arbitration phase, the Port Arbiter serves one transaction from the Ingress Port indicated by the Port Number of the current phase. When finished, it immediately advances to the next phase. A phase is skipped, i.e., the Port Arbiter simply moves to the next phase immediately if the Ingress Port indicated by the phase does not contain any transaction for the VC (note that a phase cannot contain the Egress Port's Port Number).
- When the Time-based WRR Port Arbitration is used for a VC of any given Port, at each arbitration phase aligning to a virtual timeslot, the Port Arbiter serves one transaction from the Ingress Port indicated by the Port Number of the current phase. It advances to the next phase at the next virtual timeslot. A phase indicates an "idle" timeslot, i.e., the Port Arbiter does not serve any transaction during the phase, if

the phase contains the Egress Port's Port Number, or

the Ingress Port indicated by the phase does not contain any transaction for the VC.

The Port Arbitration Table Entry Size field in the Port VC Capability register 1 determines the table entry size. The length of the table is determined by the Port Arbitration Select field as shown in Table 7-53.

When the Port Arbitration Table is used by the default Port Arbitration for the default VC, the default values for the table entries must contain at least one entry for each of the other PCI Express Ports of the component to ensure forward progress for the default VC for each Port. The table may contain RR or RR-like fair Port Arbitration for the default VC.

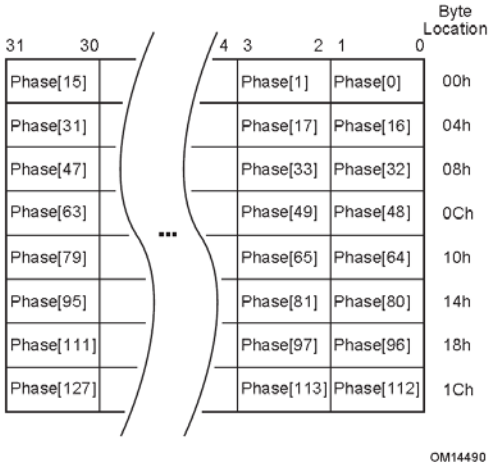


Figure 7-55: Example Port Arbitration Table with 128 Phases and 2-bit Table Entries

Table 7-53: Length of Port Arbitration Table

Port Arbitration Select	Port Arbitration Table Length (in Number of Entries)
001b	32
010b	64
011b	128
100b	128
101b	256

7.12. Device Serial Number Capability

The PCI Express Device Serial Number Capability is an optional Extended Capability that may be implemented by any PCI Express device Function. The Device Serial Number is a read-only 64-bit value that is unique for a given PCI Express device. Figure 7-56 details allocation of register fields in the PCI Express Capability structure.

All multi-Function devices that implement this Capability must implement it for Function 0; other Functions that implement this Capability must return the same Device Serial Number value as that reported by Function 0.

A PCI Express multi-device component such as a PCI Express Switch that implements this Capability must return the same Device Serial Number for each device.

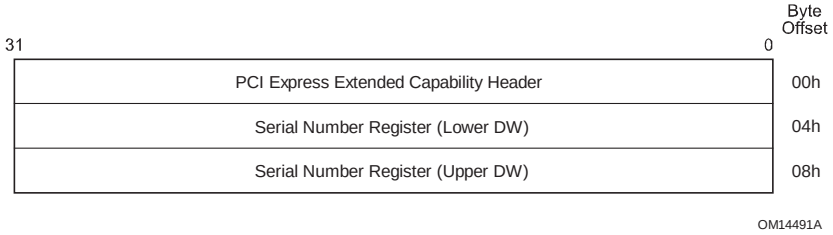


Figure 7-56: PCI Express Device Serial Number Capability Structure

7.12.1. Device Serial Number Extended Capability Header (Offset 00h)

Figure 7-57 details allocation of register fields in the Device Serial Number Extended Capability header; Table 7-54 provides the respective bit definitions. Refer to Section 7.9.3 for a description of the PCI Express Extended Capability header. The Extended Capability ID for the Device Serial Number Capability is 0003h.

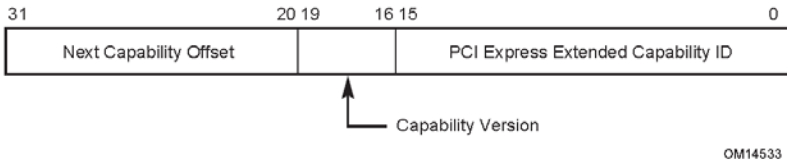


Figure 7-57: Device Serial Number Extended Capability Header

Table 7-54: Device Serial Number Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Device Serial Number Capability is 0003h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.12.2. Serial Number Register (Offset 04h)

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit extended unique identifier (EUI-64™). Figure 7-58 details allocation of register fields in the Serial Number register; Table 7-55 provides the respective bit definitions.

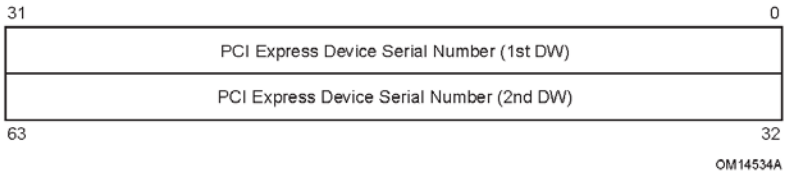


Figure 7-58: Serial Number Register

Table 7-55: Serial Number Register

Bit Location	Register Description	Attributes
63:0	**PCI Express Device Serial Number** – This field contains the IEEE defined 64-bit extended unique identifier (EUI-64™). This identifier includes a 24-bit company id value assigned by IEEE registration authority and a 40-bit extension identifier assigned by the manufacturer.	RO

7.13. PCI Express Root Complex Link Declaration Capability

The PCI Express Root Complex Link Declaration Capability is an optional Capability that is permitted to be implemented by Root Ports, Root Complex Integrated Endpoints, or RCRBs to declare a Root Complex's internal topology.

A Root Complex consists of one or more following elements:

- PCI Express Root Port
- A default system Egress Port or an internal sink unit such as memory (represented by an RCRB)
- Internal Data Paths/Links (represented by an RCRB on either side of an internal Link)
- Integrated devices
- Functions

A Root Complex Component is a logical aggregation of the above described Root Complex elements. No single element can be part of more than one Root Complex Component. Each Root Complex Component must have a unique Component ID.

A Root Complex is represented either as an opaque Root Complex or as a collection of one or more Root Complex Components.

The PCI Express Root Complex Link Declaration Capability is permitted to be present in a Root Complex element's Configuration Space or RCRB. It declares Links from the respective element to other elements of the same Root Complex Component or to an element in another Root Complex Component. The Links are required to be declared bidirectional such that each valid data path from one element to another has corresponding Link entries in the Configuration Space (or RCRB) of both elements.

The PCI Express Root Complex Link Declaration Capability is permitted to also declare an association between a Configuration Space element (Root Port or Root Complex Integrated Endpoint) and an RCRB Header Capability (see Section 7.20) contained in an RCRB that affects the behavior of the Configuration Space element. Note that an RCRB Header association is not declared bidirectional; the association is only declared by the Configuration Space element and not by the target RCRB.

IMPLEMENTATION NOTE

Topologies to Avoid

Topologies that create more than one data path between any two Root Complex elements (either directly or through other Root Complex elements) may not be able to support bandwidth allocation in a standard manner. The description of how traffic is routed through such a topology is implementation specific, meaning that general purpose-operating systems may not have enough information about such a topology to correctly support bandwidth allocation. In order to circumvent this problem, these operating systems may require that a single RCRB element (of type Internal Link) not declare more than one Link to a Root Complex Component other than the one containing the RCRB element itself.

The PCI Express Root Complex Link Declaration Capability, as shown in Figure 7-59, consists of the PCI Express Extended Capability header and Root Complex Element Self Description followed by one or more Root Complex Link Entries.

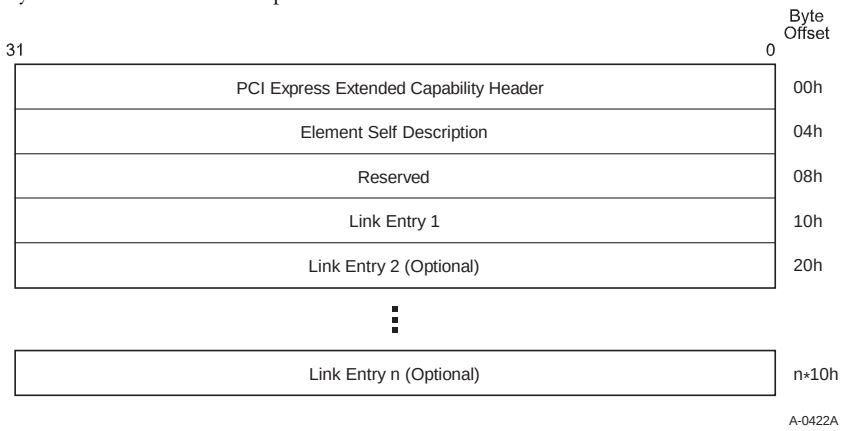


Figure 7-59: PCI Express Root Complex Link Declaration Capability

7.13.1. Root Complex Link Declaration Extended Capability Header (Offset 00h)

The Extended Capability ID for the Root Complex Link Declaration Capability is 0005h.

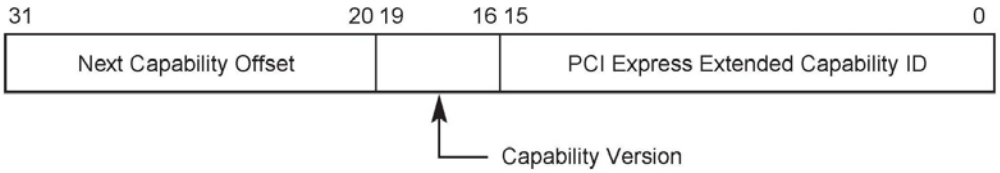


Figure 7-60: Root Complex Link Declaration Extended Capability Header

Table 7-56: Root Complex Link Declaration Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Link Declaration Capability is 0005h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits.	RO

7.13.2. Element Self Description (Offset 04h)

The Element Self Description register provides information about the Root Complex element containing the Link Declaration Capability.

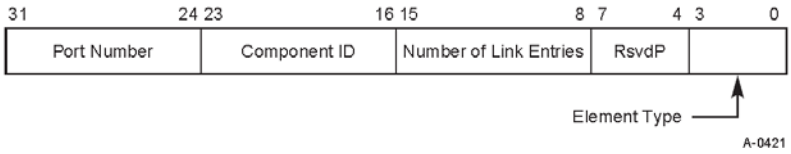


Figure 7-61: Element Self Description Register

Table 7-57: Element Self Description Register

Bit Location	Register Description	Attributes
3:0	**Element Type** – This field indicates the type of the Root Complex Element. Defined encodings are: 0h Configuration Space Element 1h System Egress Port or internal sink (memory) 2h Internal Root Complex Link 3h-Fh Reserved	RO
15:8	**Number of Link Entries** – This field indicates the number of Link entries following the Element Self Description. This field must report a value of 01h or higher.	HwInit
23:16	**Component ID** – This field identifies the Root Complex Component that contains this Root Complex Element. Component IDs must start at 01h, as a value of 00h is Reserved.	HwInit
31:24	**Port Number** – This field specifies the Port Number associated with this element with respect to the Root Complex Component that contains this element. An element with a Port Number of 00h indicates the default Egress Port to configuration software.	HwInit

7.13.3. Link Entries

Link Entries start at offset 10h of the PCI Express Root Complex Link Declaration Capability structure. Each Link Entry consists of a Link description followed by a 64-bit Link address at offset 08h from the start of Link entry identifying the target element for the declared Link. A Link Entry declares an internal Link to another Root Complex Element.

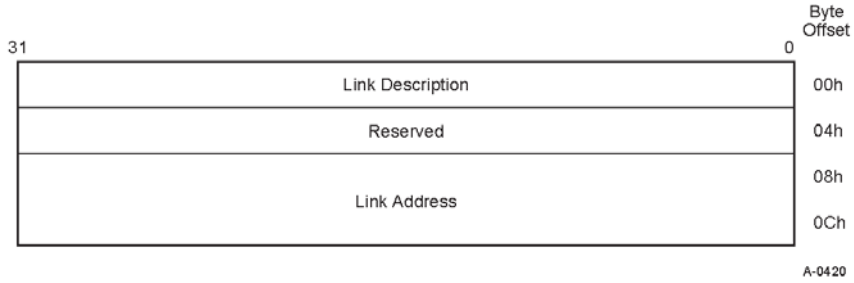


Figure 7-62: Link Entry

7.13.3.1. Link Description

The Link Description is located at offset 00h from the start of a Link Entry and is defined as follows:

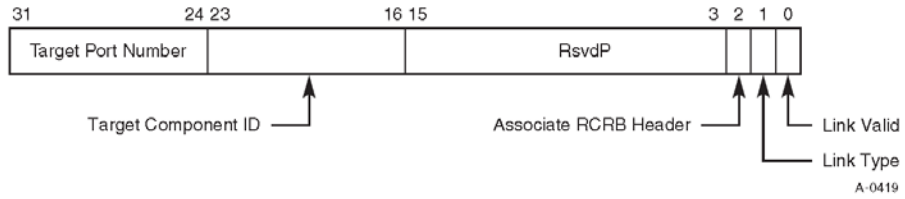


Figure 7-63: Link Description Register

Table 7-58: Link Description Register

Bit Location	Register Description	Attributes
0	**Link Valid** – When Set, this bit indicates that the Link Entry specifies a valid Link. Link entries that do not have either this bit Set or the Associate RCRB Header bit Set (or both) are ignored by software.	HwInit
1	**Link Type** – This bit indicates the target type of the Link and defines the format of the Link address field. Defined Link Type values are: 0b – Link points to memory-mapped space[117] (for RCRB). The Link address specifies the 64-bit base address of the target RCRB. 1b – Link points to Configuration Space (for a Root Port or Root Complex Integrated Endpoint). The Link address specifies the configuration address (PCI Segment Group, Bus, Device, Function) of the target element.	HwInit
2	**Associate RCRB Header** – When Set, this bit indicates that the Link Entry associates the declaring element with an RCRB Header Capability in the target RCRB. Link entries that do not have either this bit Set or the Link Valid bit Set (or both) are ignored by software. The Link Type bit must be Clear when this bit is Set.	HwInit
23:16	**Target Component ID** – This field identifies the Root Complex Component that is targeted by this Link entry. Components IDs must start at 01h, as a value of 00h is Reserved	HwInit
31:24	**Target Port Number** – This field specifies the Port Number associated with the element targeted by this Link entry; the target Port Number is with respect to the Root Complex Component (identified by the Target Component ID) that contains the target element.	HwInit

7.13.3.2. Link Address

The Link address is a HwInit field located at offset 08h from the start of a Link Entry that identifies the target element for the Link entry. For a Link of Link Type 0 in its Link Description, the Link address specifies the memory-mapped base address of RCRB. For a Link of Link Type 1 in its Link Description, the Link address specifies the Configuration Space address of a PCI Express Root Port or a Root Complex Integrated Endpoint.

7.13.3.2.1. Link Address for Link Type 0

For a Link pointing to a memory-mapped RCRB (Link Type bit = 0), the first DWORD specifies the lower 32 bits of the RCRB base address of the target element as shown below; bits 11:0 are

[117] The memory-mapped space for accessing an RCRB is not the same as Memory Space, and must not overlap with Memory Space.

hardwired to 000h and Reserved for future use. The second DWORD specifies the high order 32 bits (63:32) of the RCRB base address of the target element.

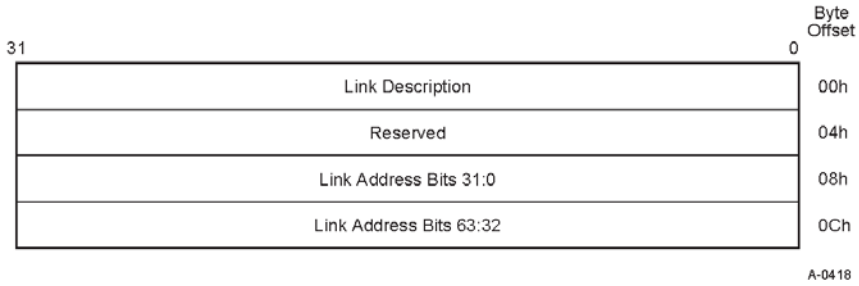


Figure 7-64: Link Address for Link Type 0

7.13.3.2.2. Link Address for Link Type 1

For a Link pointing to the Configuration Space of a Root Complex element (Link Type bit = 1), bits in the first DWORD specify the Bus, Device, and Function Number of the target element. As shown in Figure 7-65, bits 2:0 (N) encode the number of bits n associated with the Bus Number, with N = 000b specifying $n = 8$ and all other encodings specifying $n =$ <value of N>. Bits 11:3 are Reserved and hardwired to 0. Bits 14:12 specify the Function Number, and bits 19:15 specify the Device Number. Bits (19 + n):20 specify the Bus Number, with $1 \leq n \leq 8$.

Bits 31:(20 + n) of the first DWORD together with the second DWORD optionally identify the target element's hierarchy for systems implementing the PCI Express Enhanced Configuration Access Mechanism by specifying bits 63:(20 + n) of the memory-mapped Configuration Space base address of the PCI Express hierarchy associated with the targeted element; single hierarchy systems that do not implement more than one memory mapped Configuration Space are allowed to report a value of zero to indicate default Configuration Space.

A Configuration Space base address [63:(20 + n)] equal to zero indicates that the Configuration Space address defined by bits (19 + n):12 (Bus Number, Device Number, and Function Number) exists in the default PCI Segment Group; any non-zero value indicates a separate Configuration Space base address.

Software must not use n outside the context of evaluating the Bus Number and memory-mapped Configuration Space base address for this specific target element. In particular, n does not necessarily indicate the maximum Bus Number supported by the associated PCI Segment Group.

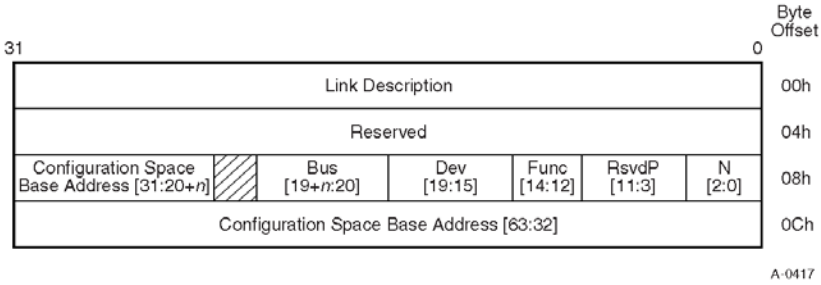


Figure 7-65: Link Address for Link Type 1

Table 7-59: Link Address for Link Type 1

Bit Location	Register Description	Attributes
2:0	**N** – Encoded number of Bus Number bits	HwInit
14:12	**Function Number**	HwInit
19:15	**Device Number**	HwInit
(19 + *n*):20	**Bus Number**	HwInit
63:(20 + *n*)	**PCI Express Configuration Space Base Address** (1 ≤ *n* ≤ 8) Note: A Root Complex that does not implement multiple Configuration Spaces is allowed to report this field as 0.	HwInit

7.14. PCI Express Root Complex Internal Link Control Capability

The PCI Express Root Complex Internal Link Control Capability is an optional Capability that controls an internal Root Complex Link between two distinct Root Complex Components. This Capability is valid for RCRBs that declare an Element Type field as Internal Root Complex Link in the Element Self-Description register of the Root Complex Link Declaration Capability structure.

The Root Complex Internal Link Control Capability structure is defined as shown in Figure 7-66.

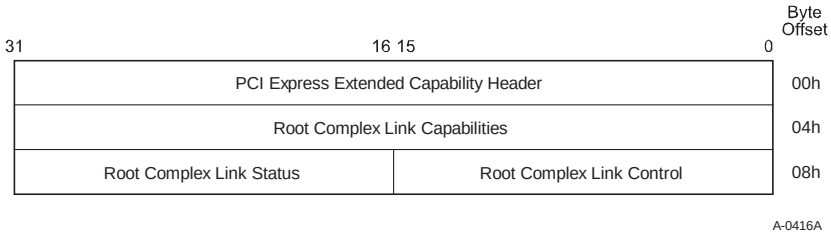


Figure 7-66: Root Complex Internal Link Control Capability

7.14.1. Root Complex Internal Link Control Extended Capability Header (Offset 00h)

The Extended Capability ID for the Root Complex Internal Link Control Capability is 0006h.

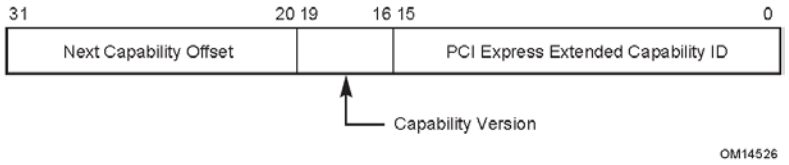


Figure 7-67: Root Internal Link Control Extended Capability Header

Table 7-60: Root Complex Internal Link Control Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Link Declaration Capability is 0006h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits.	RO

7.14.2. Root Complex Link Capabilities Register (Offset 04h)

The Root Complex Link Capabilities register identifies capabilities for this Link.

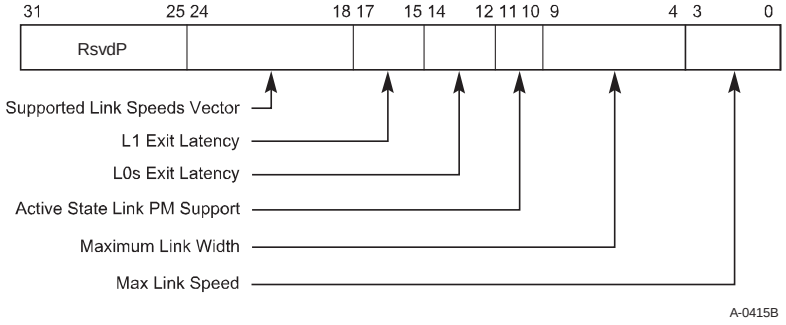


Figure 7-68: Root Complex Link Capabilities Register

Table 7-61: Root Complex Link Capabilities Register

Bit Location	Register Description	Attributes
3:0	**Max Link Speed** – This field indicates the maximum Link speed of the associated Link. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Root Complex Link Capabilities register) that corresponds to the maximum Link speed. Defined encodings are: 0001b Supported Link Speeds Vector field bit 0 0010b Supported Link Speeds Vector field bit 1 0011b Supported Link Speeds Vector field bit 2 0100b Supported Link Speeds Vector field bit 3 0101b Supported Link Speeds Vector field bit 4 0110b Supported Link Speeds Vector field bit 5 0111b Supported Link Speeds Vector field bit 6 All other encodings are reserved. A Root Complex that does not support this feature must report 0000b in this field.	RO
9:4	**Maximum Link Width** – This field indicates the maximum width of the given Link. Defined encodings are: 00 0001b x1 00 0010b x2 00 0100b x4 00 1000b x8 00 1100b x12 01 0000b x16 10 0000b x32 All other encodings are Reserved. A Root Complex that does not support this feature must report 00 0000b in this field.	RO
11:10	**Active State Power Management (ASPM) Support** – This field indicates the level of ASPM supported on the given Link. Defined encodings are: 00b No ASPM Support 01b L0s Supported 10b L1 Supported 11b L0s and L1 Supported	RO

Bit Location	Register Description	Attributes
14:12	**L0s Exit Latency** – This field indicates the L0s exit latency for the given Link. The value reported indicates the length of time this Port requires to complete transition from L0s to L0. If L0s is not supported, the value is undefined. Defined encodings are: 000b Less than 64 ns 001b 64 ns to less than 128 ns 010b 128 ns to less than 256 ns 011b 256 ns to less than 512 ns 100b 512 ns to less than 1 µs 101b 1 µs to less than 2 µs 110b 2 µs to 4 µs 111b More than 4 µs	RO
17:15	**L1 Exit Latency** – This field indicates the L1 exit latency for the given Link. The value reported indicates the length of time this Port requires to complete transition from ASPM L1 to L0. If ASPM L1 is not supported, the value is undefined. Defined encodings are: 000b Less than 1 µs 001b 1 µs to less than 2 µs 010b 2 µs to less than 4 µs 011b 4 µs to less than 8 µs 100b 8 µs to less than 16 µs 101b 16 µs to less than 32 µs 110b 32 µs to 64 µs 111b More than 64 µs	RO

Bit Location	Register Description	Attributes
24:18	Supported Link Speeds Vector – This field indicates the supported Link speed(s) of the associated Link. For each bit, a value of 1b indicates that the corresponding Link speed is supported; otherwise, the Link speed is not supported. See Section 4.3.2 for further requirements. Bit definitions within this field are: Bit 0 2.5 GT/s Bit 1 5.0 GT/s Bit 2 8.0 GT/s Bits 6:3 RsvdP For M-PCIe the bit definitions within this field are: Bit 0 HS-G1/HS-G2 Bit 1 HS-G3 Bits 6:2 RsvdP	RO

Comment [VS288]: Errata A1

Comment [VS289]: ECN: M-PCIe

IMPLEMENTATION NOTE

Supported Link Speeds With Earlier Hardware

Hardware components compliant to versions of the Base Specification prior to Revision 3.0 did not implement the Supported Link Speeds Vector field and instead returned 0000 000b in bits 24:18.

For software to determine the supported Link speeds for components where this field is contains 0000 000b, software can read bits 3:0 of the Root Complex Link Capabilities register (now defined to be the Max Link Speed field), and interpret the value as follows:

0001b 2.5 GT/s Link speed supported

0010b 5.0 GT/s and 2.5 GT/s Link speeds supported

For such components, the same encoding is also used for the values for the Current Link Speed field (in the Root Complex Link Status register).

Formatted: Font: Arial Black, 14 pt, Font color: Black, All caps

IMPLEMENTATION NOTE

Software Management of Link Speeds With Future Hardware

It is strongly encouraged that software primarily utilize the Supported Link Speeds Vector instead of the Max Link Speed field, so that software can determine the exact set of supported speeds on current and future hardware. This can avoid software being confused if a future specification defines Links that do not require support for all slower speeds.

Comment [VS290]: Errata A2

7.14.3. Root Complex Link Control Register (Offset 08h)

The Root Complex Link Control register controls parameters for this internal Link.

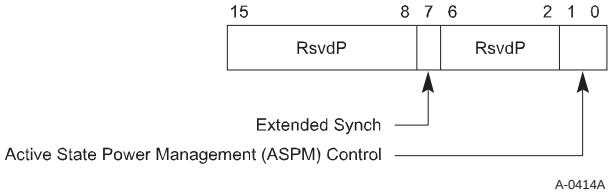


Figure 7-69: Root Complex Link Control Register

Table 7-62: Root Complex Link Control Register

Bit Location	Register Description	Attributes
1:0	**Active State Power Management (ASPM) Control** – This field controls the level of ASPM enabled on the given Link. Defined encodings are: 00b Disabled 01b L0s Entry Enabled 10b L1 Entry Enabled 11b L0s and L1 Entry Enabled Note: "L0s Entry Enabled" enables the Transmitter to enter L0s. If L0s is supported, the Receiver must be capable of entering L0s even when the Transmitter is disabled from entering L0s (00b or 10b). Default value of this field is implementation specific. Software must not enable L0s in either direction on a given Link unless components on both sides of the Link each support L0s, as indicated by their ASPM Support field values. Otherwise, the result is undefined. ASPM L1 must be enabled by software in the Upstream component on a Link prior to enabling ASPM L1 in the Downstream component on that Link. When disabling ASPM L1, software must disable ASPM L1 in the Downstream component on a Link prior to disabling ASPM L1 in the Upstream component on that Link. ASPM L1 must only be enabled on the Downstream component if both components on a Link support ASPM L1. A Root Complex that does not support this feature for the given internal Link must hardwire this field to 00b.	RW

Bit Location	Register Description	Attributes
7	**Extended Synch** – This bit when Set forces the transmission of additional Ordered Sets when exiting the L0s state (see Section 4.2.4.5) and when in the Recovery state (see Section 4.2.6.4.1). This mode provides external devices (e.g., logic analyzers) monitoring the Link time to achieve bit and Symbol lock before the Link enters the L0 state and resumes communication. A Root Complex that does not support this feature for the given internal Link must hardwire this bit to 0b. With the M-PCIe PHY, this feature is unused and this bit is hardwired to 0b. Default value for this bit is 0b.	RW

Comment [VS291]: ECN: M-PCIe

7.14.4. Root Complex Link Status Register (Offset 0Ah)

The Root Complex Link Status register provides information about Link specific parameters.

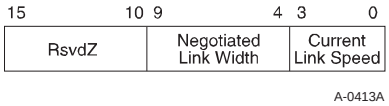


Figure 7-70: Root Complex Link Status Register

Table 7-63: Root Complex Link Status Register

Bit Location	Register Description	Attributes
3:0	**Current Link Speed** – This field indicates the negotiated Link speed of the given Link. The encoded value specifies a bit location in the Supported Link Speeds Vector (in the Root Complex Link Capabilities register) that corresponds to the current Link speed. Defined encodings are: 0001b Supported Link Speeds Vector field bit 0 0010b Supported Link Speeds Vector field bit 1 0011b Supported Link Speeds Vector field bit 2 0100b Supported Link Speeds Vector field bit 3 0101b Supported Link Speeds Vector field bit 4 0110b Supported Link Speeds Vector field bit 5 0111b Supported Link Speeds Vector field bit 6 All other encodings are Reserved. The value in this field is undefined when the Link is not up. A Root Complex that does not support this feature must report 0000b in this field.	RO

Bit Location	Register Description	Attributes
9:4	**Negotiated Link Width** – This field indicates the negotiated width of the given Link. Defined encodings are: 00 0001b x1 00 0010b x2 00 0100b x4 00 1000b x8 00 1100b x12 01 0000b x16 10 0000b x32 All other encodings are Reserved. The value in this field is undefined when the Link is not up. A Root Complex that does not support this feature must hardwire this field to 00 0000b.	RO

7.15. Power Budgeting Capability

The PCI Express Power Budgeting Capability allows the system to allocate power to devices that are added to the system at runtime. Through this Capability, a device can report the power it consumes on a variety of power rails, in a variety of device power-management states, in a variety of operating conditions. The system can use this information to ensure that the system is capable of providing the proper power and cooling levels to the device. Failure to indicate proper device power consumption may risk device or system failure.

Implementation of the Power Budgeting Capability is optional for PCI Express devices that are implemented either in a form factor which does not require Hot-Plug support, or that are integrated on the system board. PCI Express form factor specifications may require support for power budgeting. Figure 7-71 details allocation of register fields in the Power Budgeting Capability structure.

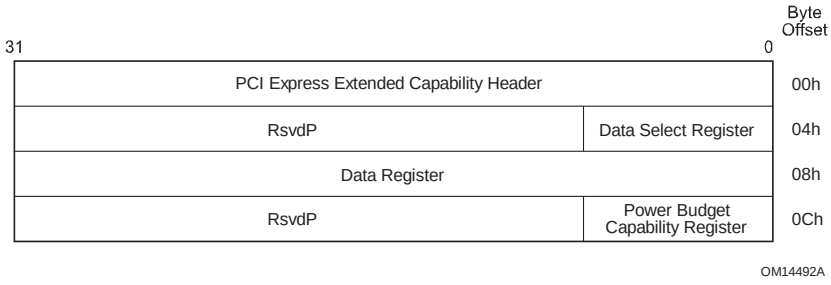


Figure 7-71: PCI Express Power Budgeting Capability Structure

7.15.1. Power Budgeting Extended Capability Header (Offset 00h)

Figure 7-72 details allocation of register fields in the Power Budgeting Extended Capability header; Table 7-64 provides the respective bit definitions. Refer to Section 7.9.3 for a description of the PCI Express Extended Capability header. The Extended Capability ID for the Power Budgeting Capability is 0004h.

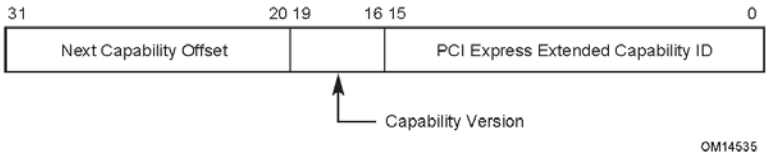


Figure 7-72: Power Budgeting Extended Capability Header

Table 7-64: Power Budgeting Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Power Budgeting Capability is 0004h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.15.2. Data Select Register (Offset 04h)

The Data Select Register is an 8-bit ~~This~~ read-write register that indexes the Power Budgeting Data reported through the Data register and selects the DWORD of Power Budgeting Data that is to appear in the Data register. ~~Index v~~Values for this register start at zero to select the first DWORD of Power Budgeting Data; subsequent DWORDs of Power Budgeting Data are selected by increasing index values. The default value of this register is undefined.

Comment [VS292]: Errata A17

7.15.3. Data Register (Offset 08h)

This read-only register returns the DWORD of Power Budgeting Data selected by the Data Select register. Each DWORD of the Power Budgeting Data describes the power usage of the device in a particular operating condition. Power Budgeting Data for different operating conditions is not required to be returned in any particular order, as long as incrementing the Data Select register causes information for a different operating condition to be returned. If the Data Select register contains a value greater than or equal to the number of operating conditions for which the device provides power information, this register must return all zeros. The default value of this register is undefined. Figure 7-73 details allocation of register fields in the Power Budgeting Data register; Table 7-65 provides the respective bit definitions.

Comment [VS293]: Errata A17

The Base Power and Data Scale fields describe the power usage of the device; the Power Rail, Type, PM State, and PM Sub State fields describe the conditions under which the device has this power usage.

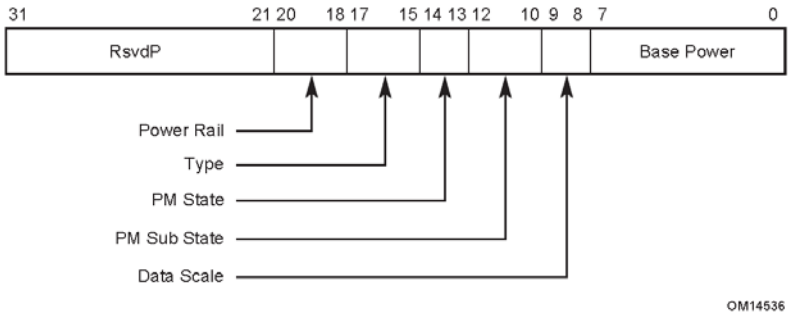


Figure 7-73: Power Budgeting Data Register

Table 7-65: Power Budgeting Data Register

Bit Location	Register Description	Attributes
7:0	**Base Power** – Specifies in watts the base power value in the given operating condition. This value must be multiplied by the data scale to produce the actual power consumption value except when the Data Scale field equals 00b (1.0x) and Base Power exceeds EFh, the following alternative encodings are used: F0h = 250 W Slot Power Limit F1h = 275 W Slot Power Limit F2h = 300 W Slot Power Limit F3h to FFh = Reserved for Slot Power Limit values above 300 W	RO

Comment [VS294]: Errata A17

Bit Location	Register Description	Attributes
9:8	**Data Scale** – Specifies the scale to apply to the Base Power value. The power consumption of the device is determined by multiplying the contents of the Base Power field with the value corresponding to the encoding returned by this field, except as noted above. Defined encodings are: 00b 1.0x 01b 0.1x 10b 0.01x 11b 0.001x	RO
12:10	**PM Sub State** – Specifies the power management sub state of the operating condition being described. Defined encodings are: 000b Default Sub State 001b – 111b Device Specific Sub State	RO
14:13	**PM State** – Specifies the power management state of the operating condition being described. Defined encodings are: 00b D0 01b D1 10b D2 11b D3 A device returns 11b in this field and Aux or PME Aux in the Type field~~register~~ to specify the D3-Cold PM State. An encoding of 11b along with any other Type field~~register~~ value specifies the D3-Hot state.	RO
17:15	**Type** – Specifies the type of the operating condition being described. Defined encodings are: 000b PME Aux 001b Auxiliary 010b Idle 011b Sustained 111b Maximum All other encodings are Reserved.	RO

Comment [VS295]: Errata A17

Bit Location	Register Description	Attributes
20:18	**Power Rail** – Specifies the thermal load or power rail of the operating condition being described. Defined encodings are: 000b Power (12V) 001b Power (3.3V) 010b Power (1.5V or 1.8V) 111b Thermal All other encodings are Reserved.	RO

A device that implements the Power Budgeting Capability is required to provide data values for the D0 Maximum and D0 Sustained PM State and Type combinations for every power rail from which it consumes power; data for the D0 Maximum and D0 Sustained for Thermal must also be provided if these values are different from the sum of the values for an operating condition reported for D0 Maximum and D0 Sustained on the power rails.

Devices that support auxiliary power or PME from auxiliary power must provide data for the appropriate power type (Auxiliary or PME Aux).

7.15.4. Power Budget Capability Register (Offset 0Ch)

This register indicates the power budgeting capabilities of a device. Figure 7-74 details allocation of register fields in the Power Budget Capability register; Table 7-66 provides the respective bit definitions.

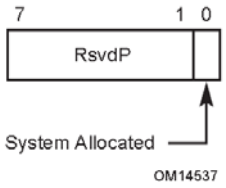


Figure 7-74: Power Budget Capability Register

Table 7-66: Power Budget Capability Register

Bit Location	Register Description	Attributes
0	**System Allocated** – When Set, this bit indicates that the power budget for the device is included within the system power budget. Reported Power Budgeting Data for this device must be ignored by software for power budgeting decisions if this bit is Set.	HwInit

7.16. ACS Extended Capability

The ACS Extended Capability is an optional capability that provides enhanced access controls (see Section 6.12). This capability may be implemented by a Root Port, a Switch Downstream Port, or a multi-Function device Function. It is never applicable to a PCI Express to PCI Bridge or Root Complex Event Collector. It is not applicable to a Switch Upstream Port unless that Switch Upstream Port is a Function in a multi-Function device.

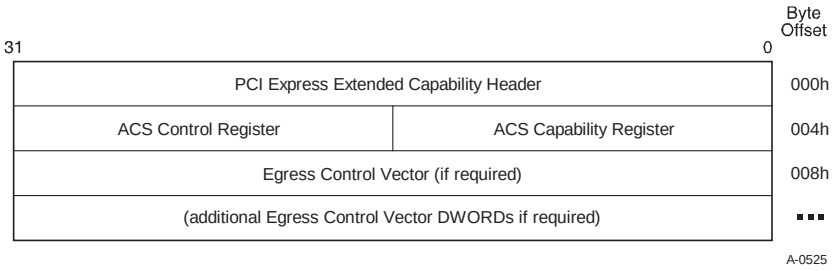


Figure 7-75: ACS Extended Capability

7.16.1. ACS Extended Capability Header (Offset 00h)

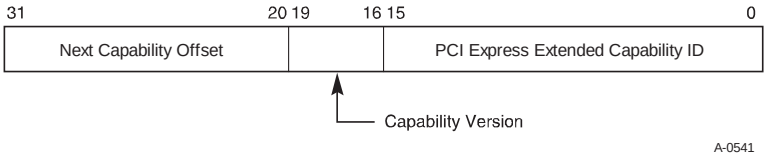


Figure 7-76: ACS Extended Capability Header

Table 7-67: ACS Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the ACS Extended Capability is 000Dh.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.16.2. ACS Capability Register (Offset 04h)

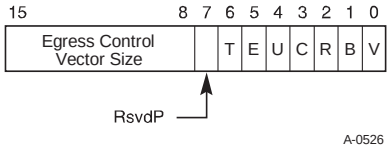


Figure 7-77: ACS Capability Register

Table 7-68: ACS Capability Register

Bit Location	Register Description	Attributes
0	**ACS Source Validation (V)** – Required for Root Ports and Switch Downstream Ports; must be hardwired to 0b otherwise. If 1b, indicates that the component implements ACS Source Validation.	RO
1	**ACS Translation Blocking (B)** – Required for Root Ports and Switch Downstream Ports; must be hardwired to 0b otherwise. If 1b, indicates that the component implements ACS Translation Blocking.	RO
2	**ACS P2P Request Redirect (R)** – Required for Root Ports that support peer-to-peer traffic with other Root Ports; required for Switch Downstream Ports; required for multi-Function device Functions that support peer-to-peer traffic with other Functions; must be hardwired to 0b otherwise. If 1b, indicates that the component implements ACS P2P Request Redirect.	RO
3	**ACS P2P Completion Redirect (C)** – Required for all Functions that support ACS P2P Request Redirect; must be hardwired to 0b otherwise. If 1b, indicates that the component implements ACS P2P Completion Redirect.	RO
4	**ACS Upstream Forwarding (U)** – Required for Root Ports if the RC supports Redirected Request Validation; required for Switch Downstream Ports; must be hardwired to 0b otherwise. If 1b, indicates that the component implements ACS Upstream Forwarding.	RO
5	**ACS P2P Egress Control (E)** – Optional for Root Ports, Switch Downstream Ports, and multi-Function device Functions; must be hardwired to 0b otherwise. If 1b, indicates that the component implements ACS P2P Egress Control.	RO
6	**ACS Direct Translated P2P (T)** – Required for Root Ports that support Address Translation Services (ATS) and also support peer-to-peer traffic with other Root Ports; required for Switch Downstream Ports; required for multi-Function device Functions that support Address Translation Services (ATS) and also support peer-to-peer traffic with other Functions; must be hardwired to 0b otherwise. If 1b, indicates that the component implements ACS Direct Translated P2P.	RO

Bit Location	Register Description	Attributes
15:8	**Egress Control Vector Size** – Encodings 01h-FFh directly indicate the number of applicable bits in the Egress Control Vector; the encoding 00h indicates 256 bits. If the ACS P2P Egress Control (E) bit is 0b, the value of the size field is undefined, and the Egress Control Vector register is not required to be present.	HwInit

7.16.3. ACS Control Register (Offset 06h)

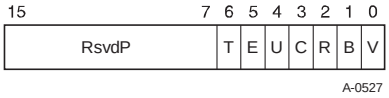


Figure 7-78: ACS Control Register

Table 7-69: ACS Control Register

Bit Location	Register Description	Attributes
0	**ACS Source Validation Enable (V)** – When Set, the component validates the Bus Number from the Requester ID of Upstream Requests against the secondary/subordinate Bus Numbers. Default value of this bit is 0b. Must be hardwired to 0b if the ACS Source Validation functionality is not implemented.	RW
1	**ACS Translation Blocking Enable (B)** – When Set, the component blocks all Upstream Memory Requests whose Address Translation (AT) field is not set to the default value. Default value of this bit is 0b. Must be hardwired to 0b if the ACS Translation Blocking functionality is not implemented.	RW
2	**ACS P2P Request Redirect Enable (R)** – In conjunction with ACS P2P Egress Control and ACS Direct Translated P2P mechanisms, determines when the component redirects peer-to-peer Requests Upstream (see Section 6.12.3). Note that with Downstream Ports, this bit only applies to Upstream Requests arriving at the Downstream Port, and whose normal routing targets a different Downstream Port. Default value of this bit is 0b. Must be hardwired to 0b if the ACS P2P Request Redirect functionality is not implemented.	RW
3	**ACS P2P Completion Redirect Enable (C)** – Determines when the component redirects peer-to-peer Completions Upstream; applicable only to Completions[118] whose Relaxed Ordering Attribute is clear. Default value of this bit is 0b. Must be hardwired to 0b if the ACS P2P Completion Redirect functionality is not implemented.	RW

[118] This includes Read Completions, AtomicOp Completions, and other Completions with or without Data.

Bit Location	Register Description	Attributes
4	**ACS Upstream Forwarding Enable (U)** – When Set, the component forwards Upstream any Request or Completion TLPs it receives that were redirected Upstream by a component lower in the hierarchy. Note that this bit only applies to Upstream TLPs arriving at a Downstream Port, and whose normal routing targets the same Downstream Port. Default value of this bit is 0b. Must be hardwired to 0b if the ACS Upstream Forwarding functionality is not implemented.	RW
5	**ACS P2P Egress Control Enable (E)** – In conjunction with the Egress Control Vector plus the ACS P2P Request Redirect and ACS Direct Translated P2P mechanisms, determines when to allow, disallow, or redirect peer-to-peer Requests (see Section 6.12.3). Default value of this bit is 0b. Must be hardwired to 0b if the ACS P2P Egress Control functionality is not implemented.	RW
6	**ACS Direct Translated P2P Enable (T)** – When Set, overrides the ACS P2P Request Redirect and ACS P2P Egress Control mechanisms with peer-to-peer Memory Requests whose Address Translation (AT) field indicates a Translated address (see Section 6.12.3). This bit is ignored if ACS Translation Blocking Enable (B) is 1b. Default value of this bit is 0b. Must be hardwired to 0b if the ACS Direct Translated P2P functionality is not implemented.	RW

7.16.4. Egress Control Vector (Offset 08h)

The Egress Control Vector is a read-write register that contains a bit-array. The number of bits in the register is specified by the Egress Control Vector Size field, and the register spans multiple DWORDs if required. If the ACS P2P Egress Control bit in the ACS Capability register is 0b, the Egress Control Vector Size field is undefined and the Egress Control Vector register is not required to be present.

For the general case of an Egress Control Vector spanning multiple DWORDs, the DWORD offset and bit number within that DWORD for a given arbitrary bit *K* are specified by the formulas[119]:

DWORD offset = 08h + (K div~~[120]~~ 32) * 4

DWORD bit# = K mod~~[121]~~ 32

Bits in a DWORD beyond those specified by the Egress Control Vector Size field are RsvdP.

For Root Ports and Switch Downstream Ports, each bit in the bit-array always corresponds to a Port Number. Otherwise, for Functions[122] within a multi-Function device, each bit in the bit-array corresponds to one or more Function Numbers, or a Function Group Number. For example, access to Function 2 is controlled by bit number 2 in the bit-array. For both Port Number cases and

Comment [VS296]: Errata B32 (Part C list resolved 10/13)

[119] Div is an integer divide with truncation. Mod is the remainder from an integer divide.

~~[120] Div is an integer divide with truncation.~~

~~[121] Mod is the remainder from an integer divide.~~

[122] Including Switch Upstream Ports.

Function Number cases, the bit corresponding to the Function that implements this Extended Capability structure must be hardwired to 0b.[123]

If an ARI Device implements ACS Function Groups, its Egress Control Vector Size is required to be a power-of-2 from 8 to 256, and all of its implemented Egress Control Vector bits must be RW. With ARI Devices, multiple Functions can be associated with a single bit, so for each Function, its associated bit determines how Requests from it targeting other Functions (if any) associated with the same bit are handled.

If ACS Function Groups are enabled in an ARI Device, the first 8 Egress Control Vector bits in each Function are associated with Function Group Numbers instead of Function Numbers. In this case, access control is enforced between Function Groups instead of Functions, and any implemented Egress Control Vector bits beyond the first 8 are unused.

Independent of whether an ARI Device implements ACS Function Groups, its Egress Control Vector Size is not required to cover the entire Function Number range of all Functions implemented by the Device. If ACS Function Groups are not enabled, Function Numbers are mapped to implemented Egress Control Vector bits by taking the modulo of the Egress Control Vector Size, which is constrained to be a power-of-2.

With RCs, some Port Numbers may refer to internal Ports instead of Root Ports. For Root Ports in such RCs, each bit in the bit-array that corresponds to an internal Port must be hardwired to 0b.



Figure 7-79: Egress Control Vector Register

Table 7-70: Egress Control Vector

Bit Location	Register Description	Attributes
N-1:0	**Egress Control Vector** – An N-bit bit-array configured by software, where N is given by the value in the Egress Control Vector Size field. When a given bit is set, peer-to-peer Requests targeting the associated Port, Function, or Function Group are blocked or redirected (if enabled) (see Section 6.12.3). Default value of each bit is 0b.	RW

The following examples illustrate how the vector might be configured:

- For an 8-Port Switch, each Port will have a separate vector indicating which Downstream Egress Ports it may forward Requests to.

Port 1 being not allowed to communicate with any other Downstream Ports would be configured as: 1111 1100b with bit 0 corresponding to the Upstream Port (hardwired to 0b) and bit 1 corresponding to the Ingress Port (hardwired to 0b).

Port 2 being allowed to communicate with Ports 3, 5, and 7 would be configured as: 0101 0010b.

[123] For ARI Devices, the bit must be RW. See subsequent description.

- For a 4-Function device, each Function will have a separate vector that indicates which Function it may forward Requests to.

Function 0 being not allowed to communicate with any other Functions would be configured as: 1110b with bit 0 corresponding to Function 0 (hardwired to 0b).

Function 1 being allowed to communicate with Functions 2 and 3 would be configured as: 0001b with bit 1 corresponding to Function 1 (hardwired to 0b).

7.17. PCI Express Root Complex Event Collector Endpoint Association Capability

The PCI Express Root Complex Event Collector Endpoint Association Capability is implemented by Root Complex Event Collectors.

It declares the Root Complex Integrated Endpoints supported by the Root Complex Event Collector on the same Logical Bus on which the Root Complex Event Collector is located. A Root Complex Event Collector must implement the Root Complex Event Collector Endpoint Association Capability; no other PCI Express device Function is permitted to implement this Capability.

The PCI Express Root Complex Event Collector Endpoint Association Capability, as shown in Figure 7-80, consists of the PCI Express Extended Capability header followed by a DWORD bitmap enumerating Root Complex Integrated Endpoints associated with the Root Complex Event Collector.

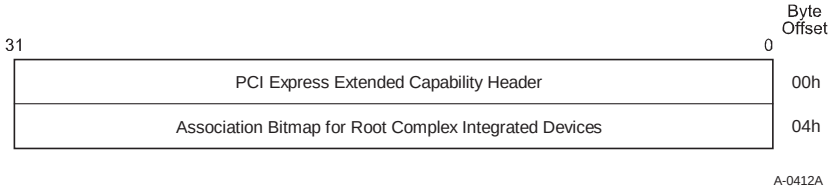


Figure 7-80: Root Complex Event Collector Endpoint Association Capability

7.17.1. Root Complex Event Collector Endpoint Association Extended Capability Header (Offset 00h)

The Extended Capability ID for the Root Complex Event Collector Endpoint Association Capability is 0007h. Figure 7-81 details allocation of fields in the Root Complex Event Collector Endpoint Association Extended Capability header; Table 7-71 provides the respective bit definitions.

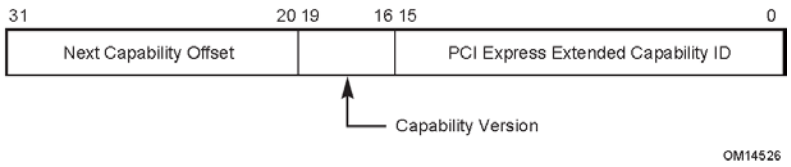


Figure 7-81: Root Complex Event Collector Endpoint Association Extended Capability Header

Table 7-71: Root Complex Event Collector Endpoint Association Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. The Extended Capability ID for the Root Complex Event Collector Endpoint Association Capability is 0007h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits.	RO

7.17.2. Association Bitmap for Root Complex Integrated Endpoints (Offset 04h)

The Association Bitmap for Root Complex Integrated Endpoints is a read-only register that sets the bits corresponding to the Device Numbers of Root Complex Integrated Endpoints supported by the Root Complex Event Collector on the same Logical Bus as the Event Collector itself. The bit corresponding to the Device Number of the Root Complex Event Collector must always be Set.

7.18. Multi-Function Virtual Channel Capability

The Multi-Function Virtual Channel (MFVC) Capability is an optional Extended Capability that permits enhanced QoS management in a multi-Function device, including TC/VC mapping, optional VC arbitration, and optional Function arbitration for Upstream Requests. When implemented, the MFVC Capability structure must be present in the Extended Configuration Space of Function 0 of the multi-Function device's Upstream Port. Figure 7-82 provides a high level view of the MFVC Capability structure. This MFVC Capability structure controls Virtual Channel assignment at the PCI Express Upstream Port of the multi-Function device, while a VC Capability structure, if present in a Function, controls the Virtual Channel assignment for that individual Function.

A multi-Function device is permitted to have an MFVC Capability structure even if none of its Functions have a VC Capability structure. However, an MFVC Capability structure is permitted only in Function 0 in the Upstream Port of a multi-Function device.

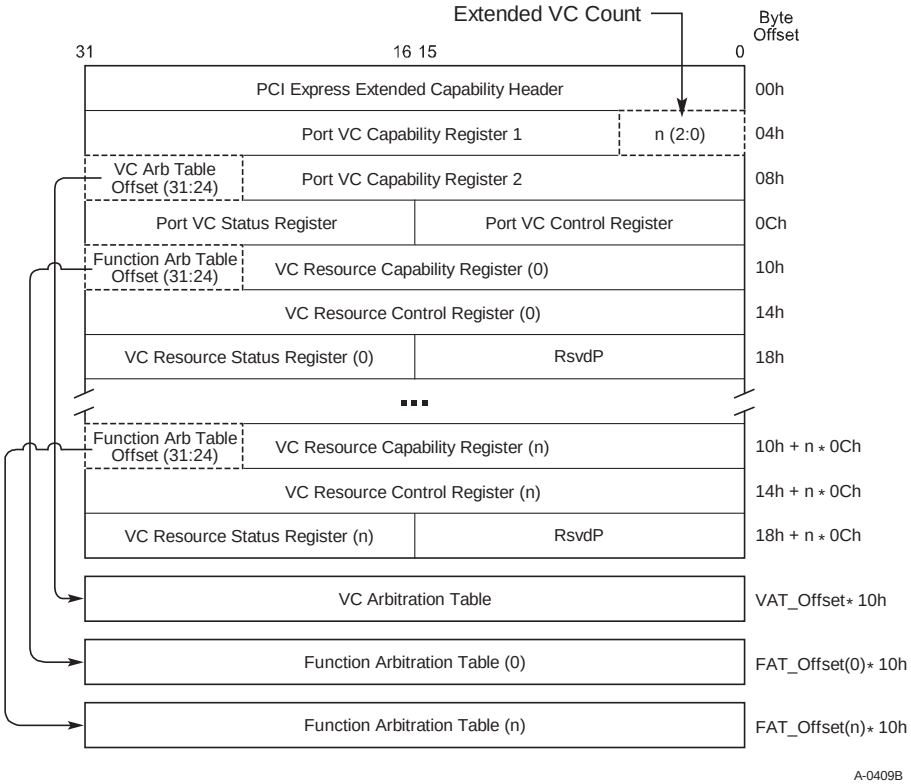


Figure 7-82: PCI Express MFVC Capability Structure

The following sections describe the registers/fields of the PCI Express MFVC Capability structure.

7.18.1. MFVC Extended Capability Header (Offset 00h)

Refer to Section 7.9.3 for a description of the PCI Express Extended Capability header. The Extended Capability ID for the MFVC Capability is 0008h. Figure 7-83 details allocation of register fields in the MFVC Extended Capability header; Table 7-72 provides the respective bit definitions.

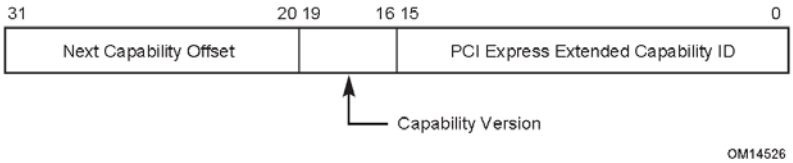


Figure 7-83: MFVC Extended Capability Header

Table 7-72: MFVC Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. The Extended Capability ID for the MFVC Capability is 0008h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.18.2. Port VC Capability Register 1 (Offset 04h)

The Port VC Capability register 1 describes the configuration of the Virtual Channels associated with a PCI Express Port of the multi-Function device. Figure 7-84 details allocation of register fields in the Port VC Capability register 1; Table 7-73 provides the respective bit definitions.

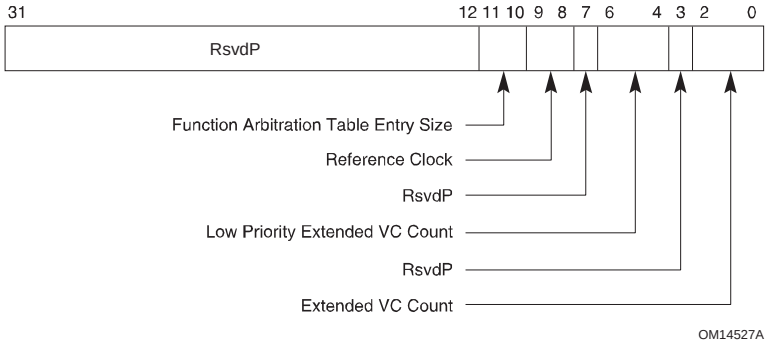


Figure 7-84: Port VC Capability Register 1

Table 7-73: Port VC Capability Register 1

Bit Location	Register Description	Attributes
2:0	**Extended VC Count** – Indicates the number of (extended) Virtual Channels in addition to the default VC supported by the device. The minimum value of this field is zero (for devices that only support the default VC). The maximum value is seven.	RO
6:4	**Low Priority Extended VC Count** – Indicates the number of (extended) Virtual Channels in addition to the default VC belonging to the low-priority VC (LPVC) group that has the lowest priority with respect to other VC resources in a strict-priority VC Arbitration. The minimum value of this field is 000b and the maximum value is Extended VC Count.	RO
9:8	**Reference Clock** – Indicates the reference clock for Virtual Channels that support time-based WRR Function Arbitration. Defined encodings are: 00b 100 ns reference clock 01b – 11b Reserved	RO
11:10	**Function Arbitration Table Entry Size** – Indicates the size (in bits) of Function Arbitration table entry in the device. Defined encodings are: 00b Size of Function Arbitration table entry is 1 bit 01b Size of Function Arbitration table entry is 2 bits 10b Size of Function Arbitration table entry is 4 bits 11b Size of Function Arbitration table entry is 8 bits	RO

7.18.3. Port VC Capability Register 2 (Offset 08h)

The Port VC Capability register 2 provides further information about the configuration of the Virtual Channels associated with a PCI Express Port of the multi-Function device. Figure 7-85 details allocation of register fields in the Port VC Capability register 2; Table 7-74 provides the respective bit definitions.



Figure 7-85: Port VC Capability Register 2

Table 7-74: Port VC Capability Register 2

Bit Location	Register Description	Attributes
7:0	**VC Arbitration Capability** – Indicates the types of VC Arbitration supported by the device for the LPVC group. This field is valid for all devices that report a Low Priority Extended VC Count greater than 0. Each bit location within this field corresponds to a VC Arbitration Capability defined below. When more than 1 bit in this field is Set, it indicates that the device can be configured to provide different VC arbitration services. Defined bit positions are: Bit 0 – Hardware fixed arbitration scheme, e.g., Round Robin Bit 1 – Weighted Round Robin (WRR) arbitration with 32 phases Bit 2 – WRR arbitration with 64 phases Bit 3 – WRR arbitration with 128 phases Bits 4-7 – Reserved	RO
31:24	**VC Arbitration Table Offset** – Indicates the location of the VC Arbitration Table. This field contains the zero-based offset of the table in DQWORDS (16 bytes) from the base address of the MFVC Capability structure. A value of 00h indicates that the table is not present.	RO

7.18.4. Port VC Control Register (Offset 0Ch)

Figure 7-86 details allocation of register fields in the Port VC Control register; Table 7-75 provides the respective bit definitions.

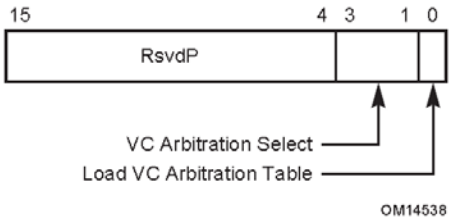


Figure 7-86: Port VC Control Register

Table 7-75: Port VC Control Register

Bit Location	Register Description	Attributes
0	**Load VC Arbitration Table** – Used by software to update the VC Arbitration Table. This bit is valid when the selected VC Arbitration uses the VC Arbitration Table. Software Sets this bit to request hardware to apply new values programmed into VC Arbitration Table; Clearing this bit has no effect. Software checks the VC Arbitration Table Status bit to confirm that new values stored in the VC Arbitration Table are latched by the VC arbitration logic. This bit always returns 0b when read.	RW
3:1	**VC Arbitration Select** – Used by software to configure the VC arbitration by selecting one of the supported VC Arbitration schemes indicated by the VC Arbitration Capability field in the Port VC Capability register 2. The permissible values of this field are numbers corresponding to one of the asserted bits in the VC Arbitration Capability field. This field cannot be modified when more than one VC in the LPVC group is enabled.	RW

7.18.5. Port VC Status Register (Offset 0Eh)

The Port VC Status register provides status of the configuration of Virtual Channels associated with a Port of the multi-Function device. Figure 7-87 details allocation of register fields in the Port VC Status register; Table 7-76 provides the respective bit definitions.

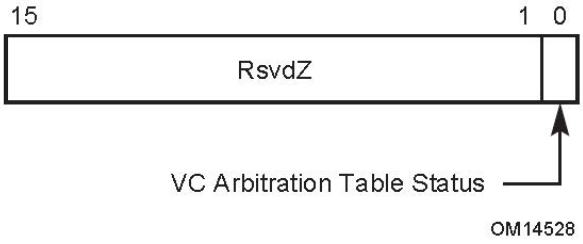


Figure 7-87: Port VC Status Register

Table 7-76: Port VC Status Register

Bit Location	Register Description	Attributes
0	**VC Arbitration Table Status** – Indicates the coherency status of the VC Arbitration Table. This bit is valid when the selected VC uses the VC Arbitration Table. This bit is Set by hardware when any entry of the VC Arbitration Table is written by software. This bit is Cleared by hardware when hardware finishes loading values stored in the VC Arbitration Table after software sets the Load VC Arbitration Table bit in the Port VC Control register. Default value of this bit is 0b.	RO

7.18.6. VC Resource Capability Register

The VC Resource Capability register describes the capabilities and configuration of a particular Virtual Channel resource. Figure 7-88 details allocation of register fields in the VC Resource Capability register; Table 7-77 provides the respective bit definitions.

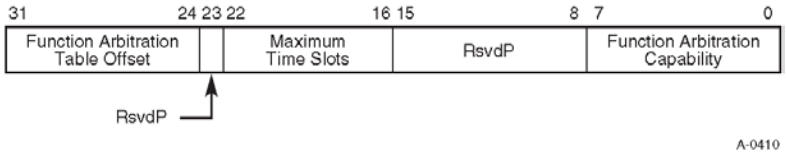


Figure 7-88: VC Resource Capability Register

Table 7-77: VC Resource Capability Register

Bit Location	Register Description	Attributes
7:0	**Function Arbitration Capability** – Indicates types of Function Arbitration supported by the VC resource. Each bit location within this field corresponds to a Function Arbitration Capability defined below. When more than 1 bit in this field is Set, it indicates that the VC resource can be configured to provide different arbitration services. Software selects among these capabilities by writing to the Function Arbitration Select field (see Section 7.18.7). Defined bit positions are: Bit 0 – Non-configurable hardware-fixed arbitration scheme, e.g., Round Robin (RR) Bit 1 – Weighted Round Robin (WRR) arbitration with 32 phases Bit 2 – WRR arbitration with 64 phases Bit 3 – WRR arbitration with 128 phases Bit 4 – Time-based WRR with 128 phases Bit 5 – WRR arbitration with 256 phases Bits 6-7 – Reserved	RO
22:16	**Maximum Time Slots** – Indicates the maximum number of time slots (minus 1) that the VC resource is capable of supporting when it is configured for time-based WRR Function Arbitration. For example, a value of 000 0000b in this field indicates the supported maximum number of time slots is 1 and a value of 111 1111b indicates the supported maximum number of time slots is 128. This field is valid only when the Function Arbitration Capability indicates that the VC resource supports time-based WRR Function Arbitration.	HwInit
31:24	**Function Arbitration Table Offset** – Indicates the location of the Function Arbitration Table associated with the VC resource. This field contains the zero-based offset of the table in DQWORDS (16 bytes) from the base address of the MFVC Capability structure. A value of 00h indicates that the table is not present.	RO

7.18.7. VC Resource Control Register

Figure 7-89 details allocation of register fields in the VC Resource Control register; Table 7-78 provides the respective bit definitions.

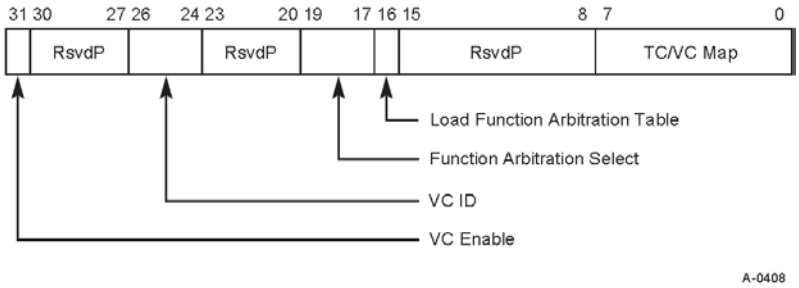


Figure 7-89: VC Resource Control Register

Table 7-78: VC Resource Control Register

Bit Location	Register Description	Attributes
7:0	**TC/VC Map** – This field indicates the TCs that are mapped to the VC resource. Bit locations within this field correspond to TC values. For example, when bit 7 is Set in this field, TC7 is mapped to this VC resource. When more than 1 bit in this field is Set, it indicates that multiple TCs are mapped to the VC resource. In order to remove one or more TCs from the TC/VC Map of an enabled VC, software must ensure that no new or outstanding transactions with the TC labels are targeted at the given Link. Default value of this field is FFh for the first VC resource and is 00h for other VC resources. Note: Bit 0 of this field is read-only. It must be hardwired to 1b for the default VC0 and hardwired to 0b for all other enabled VCs.	RW (see the note for exceptions)

Bit Location	Register Description	Attributes
16	**Load Function Arbitration Table** – When Set, this bit updates the Function Arbitration logic from the Function Arbitration Table for the VC resource. This bit is only valid when the Function Arbitration Table is used by the selected Function Arbitration scheme (that is indicated by a Set bit in the Function Arbitration Capability field selected by Function Arbitration Select). Software sets this bit to signal hardware to update Function Arbitration logic with new values stored in the Function Arbitration Table; clearing this bit has no effect. Software uses the Function Arbitration Table Status bit to confirm whether the new values of Function Arbitration Table are completely latched by the arbitration logic. This bit always returns 0b when read. Default value of this bit is 0b.	RW
19:17	**Function Arbitration Select** – This field configures the VC resource to provide a particular Function Arbitration service. The permissible value of this field is a number corresponding to one of the asserted bits in the Function Arbitration Capability field of the VC resource.	RW
26:24	**VC ID** – This field assigns a VC ID to the VC resource (see note for exceptions). This field cannot be modified when the VC is already enabled. Note: For the first VC resource (default VC), this field is a read-only field that must be hardwired to 000b.	RW

Bit Location	Register Description	Attributes
31	**VC Enable** – When Set, this bit enables a Virtual Channel (see note 1 for exceptions). The Virtual Channel is disabled when this bit is cleared. Software must use the VC Negotiation Pending bit to check whether the VC negotiation is complete. Default value of this bit is 1b for the first VC resource and 0b for other VC resource(s). Notes: 1. This bit is hardwired to 1b for the default VC (VC0), i.e., writing to this field has no effect for VC0. 2. To enable a Virtual Channel, the VC Enable bits for that Virtual Channel must be Set in both components on a Link. 3. To disable a Virtual Channel, the VC Enable bits for that Virtual Channel must be Cleared in both components on a Link. 4. Software must ensure that no traffic is using a Virtual Channel at the time it is disabled. 5. Software must fully disable a Virtual Channel in both components on a Link before re-enabling the Virtual Channel.	RW

7.18.8. VC Resource Status Register

Figure 7-90 details allocation of register fields in the VC Resource Status register; Table 7-79 provides the respective bit definitions.

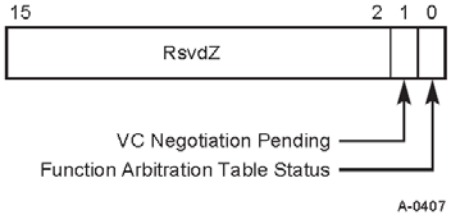


Figure 7-90: VC Resource Status Register

Table 7-79: VC Resource Status Register

Bit Location	Register Description	Attributes
0	**Function Arbitration Table Status** – This bit indicates the coherency status of the Function Arbitration Table associated with the VC resource. This bit is valid only when the Function Arbitration Table is used by the selected Function Arbitration for the VC resource. This bit is Set by hardware when any entry of the Function Arbitration Table is written to by software. This bit is Cleared by hardware when hardware finishes loading values stored in the Function Arbitration Table after software sets the Load Function Arbitration Table bit. Default value of this bit is 0b.	RO
1	**VC Negotiation Pending** – This bit indicates whether the Virtual Channel negotiation (initialization or disabling) is in pending state. When this bit is Set by hardware, it indicates that the VC resource is still in the process of negotiation. This bit is Cleared by hardware after the VC negotiation is complete. For a non-default Virtual Channel, software may use this bit when enabling or disabling the VC. For the default VC, this bit indicates the status of the process of Flow Control initialization. Before using a Virtual Channel, software must check whether the VC Negotiation Pending bits for that Virtual Channel are Clear in both components on a Link.	RO

7.18.9. VC Arbitration Table

The definition of the VC Arbitration Table in the MFVC Capability structure is identical to that in the VC Capability structure (see Section 7.11.9).

7.18.10. Function Arbitration Table

The Function Arbitration Table register in the MFVC Capability structure takes the same form as the Port Arbitration Table register in the VC Capability structure (see Section 7.11.10).

The Function Arbitration Table register is a read-write register array that is used to store the WRR or time-based WRR arbitration table for Function Arbitration for the VC resource. It is only present when one or more asserted bits in the Function Arbitration Capability field indicate that the multi-Function device supports a Function Arbitration scheme that uses a programmable arbitration table. Furthermore, it is only valid when one of the above-mentioned bits in the Function Arbitration Capability field is selected by the Function Arbitration Select field.

The Function Arbitration Table represents one Function arbitration period. Each table entry containing a Function Number or Function Group[124] Number corresponds to a phase within a Function Arbitration period. The table entry size requirements are as follows:

- The table entry size for non-ARI devices must support enough values to specify all implemented Functions plus at least one value that does not correspond to an implemented Function. For example, a table with 2-bit entries can be used by a multi-Function device with up to three Functions.
- The table entry size for ARI Devices must be either 4 bits or 8 bits.
 - If MFVC Function Groups are enabled, each entry maps to a single Function Group. Arbitration between multiple Functions within a Function Group is implementation specific, but must guarantee forward progress.
 - If MFVC Function Groups are not enabled and 4-bit entries are implemented, a given entry maps to all Functions whose Function Number modulo 8 matches its value. Similarly, if 8-bit entries are implemented, a given entry maps to all Functions whose Function Number modulo 128 matches its value. If a given entry maps to multiple Functions, arbitration between those Functions is implementation specific, but must guarantee forward progress.

A Function Number or Function Group Number written to a table entry indicates that the phase within the Function Arbitration period is assigned to the selected Function or Function Group (the Function Number or Function Group Number must be a valid one).

- When the WRR Function Arbitration is used for a VC of the Egress Port of the multi-Function device, at each arbitration phase the Function Arbiter serves one transaction from the Function or Function Group indicated by the Function Number or Function Group Number of the current phase. When finished, it immediately advances to the next phase. A phase is skipped, i.e., the Function Arbiter simply moves to the next phase immediately if the Function or Function Group indicated by the phase does not contain any transaction for the VC.
- When the Time-based WRR Function Arbitration is used for a VC of the Egress Port of the multi-Function device, at each arbitration phase aligning to a virtual timeslot, the Function Arbiter serves one transaction from the Function or Function Group indicated by the Function Number or Function Group Number of the current phase. It advances to the next phase at the next virtual timeslot. A phase indicates an "idle" timeslot, i.e., the Function Arbiter does not serve any transaction during the phase, if
 - the phase contains the Number of a Function or a Function Group that does not exist, or
 - the Function or Function Group indicated by the phase does not contain any transaction for the VC.

The Function Arbitration Table Entry Size field in the Port VC Capability register 1 determines the table entry size. The length of the table is determined by the Function Arbitration Select field as shown in Table 7-80.

[124] If an ARI Device supports MFVC Function Groups capability and ARI-aware software enables it, arbitration is based on Function Groups instead of Functions. See Section 7.23.

When the Function Arbitration Table is used by the default Function Arbitration for the default VC, the default values for the table entries must contain at least one entry for each of the active Functions or Function Groups in the multi-Function device to ensure forward progress for the default VC for the multi-Function device's Upstream Port. The table may contain RR or RR-like fair Function Arbitration for the default VC.

Table 7-80: Length of Function Arbitration Table

Function Arbitration Select	Function Arbitration Table Length (in Number of Entries)
001b	32
010b	64
011b	128
100b	128
101b	256

7.19. Vendor-Specific Capability

The PCI Express Vendor-Specific Extended Capability (VSEC) is an optional Extended Capability that is permitted to be implemented by any PCI Express Function or RCRB. This allows PCI Express component vendors to use the Extended Capability mechanism to expose vendor-specific registers.

A single PCI Express Function or RCRB is permitted to contain multiple VSEC structures.

An example usage is a set of vendor-specific features that are intended to go into an on-going series of components from that vendor. A VSEC structure can tell vendor-specific software which features a particular component supports, including components developed after the software was released.

Figure 7-91 details allocation of register fields in the VSEC structure. The structure of the PCI Express Extended Capability header and the Vendor-Specific header is architected by this specification.

With a PCI Express Function, the structure and definition of the Vendor-Specific Registers area is determined by the vendor indicated by the Vendor ID field located at byte offset 00h in PCI-compatible Configuration Space. With an RCRB, a VSEC is permitted only if the RCRB also contains an RCRB Header Capability structure, which contains a Vendor ID field indicating the vendor.

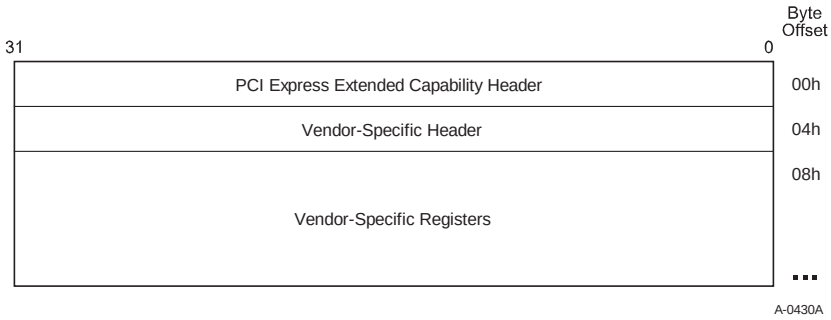


Figure 7-91: PCI Express VSEC Structure

7.19.1. Vendor-Specific Extended Capability Header (Offset 00h)

Figure 7-92 details allocation of register fields in the Vendor-Specific Extended Capability header; Table 7-81 provides the respective bit definitions. Refer to Section 7.9.3 for a description of the PCI Express Extended Capability header. The Extended Capability ID for the Vendor-Specific Capability is 000Bh.

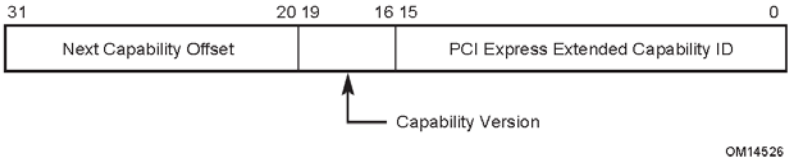


Figure 7-92: Vendor-Specific Extended Capability Header

Table 7-81: Vendor-Specific Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Vendor-Specific Capability is 000Bh.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO

Bit Location	Register Description	Attributes
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI-compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.19.2. Vendor-Specific Header (Offset 04h)

Figure 7-93 details allocation of register fields in the Vendor-Specific header; Table 7-82 provides the respective bit definitions.

Vendor-specific software must qualify the associated Vendor ID of the PCI Express Function or RCRB before attempting to interpret the values in the VSEC ID or VSEC Rev fields.

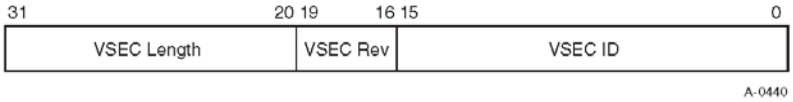


Figure 7-93: Vendor-Specific Header

Table 7-82: Vendor-Specific Header

Bit Location	Register Description	Attributes
15:0	**VSEC ID** – This field is a vendor-defined ID number that indicates the nature and format of the VSEC structure. Software must qualify the Vendor ID before interpreting this field.	RO
19:16	**VSEC Rev** – This field is a vendor-defined version number that indicates the version of the VSEC structure. Software must qualify the Vendor ID and VSEC ID before interpreting this field.	RO
31:20	**VSEC Length** – This field indicates the number of bytes in the entire VSEC structure, including the PCI Express Extended Capability header, the Vendor-Specific header, and the Vendor-Specific registers.	RO

7.20. RCRB Header Capability

The PCI Express RCRB Header Capability is an optional Extended Capability that may be implemented in an RCRB to provide a Vendor ID and Device ID for the RCRB and to permit the management of parameters that affect the behavior of Root Complex functionality associated with the RCRB.

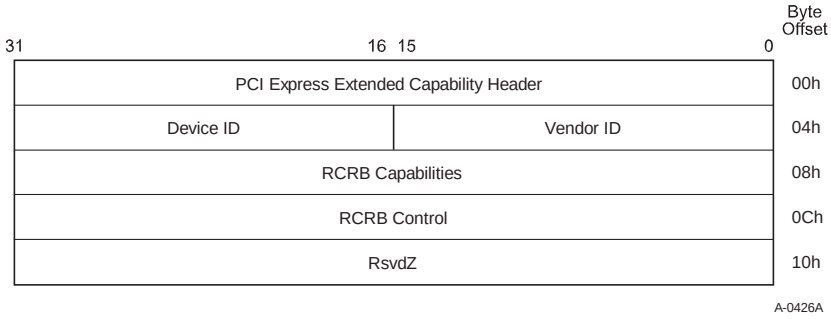


Figure 7-94: Root Complex Features Capability Structure

7.20.1. RCRB Header Extended Capability Header (Offset 00h)

Figure 7-95 details allocation of register fields in the RCRB Header Extended Capability header. Table 7-83 provides the respective bit definitions. Refer to Section 7.9.3 for a description of the PCI Express Enhanced Capabilities header. The Extended Capability ID for the RCRB Header Capability is 000Ah.

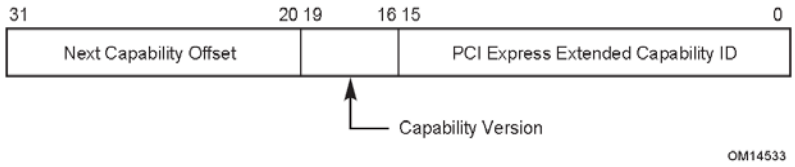


Figure 7-95: RCRB Header Extended Capability Header

Table 7-83: RCRB Header Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the RCRB Header Capability is 000Ah.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.20.2. Vendor ID (Offset 04h) and Device ID (Offset 06h)

Figure 7-96 details allocation of register fields in the RCRB Capabilities register; Table 7-84 provides the respective bit definitions.

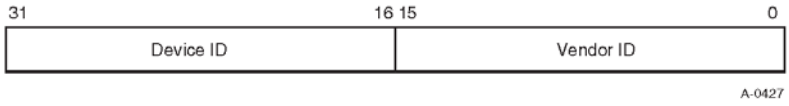


Figure 7-96: Vendor ID and Device ID

Table 7-84: Vendor ID and Device ID

Bit Location	Register Description	Attributes
15:0	**Vendor ID** – PCI-SIG assigned. Analogous to the equivalent field in PCI-compatible Configuration Space. This field provides a means to associate an RCRB with a particular vendor.	RO
31:16	**Device ID** – Vendor assigned. Analogous to the equivalent field in PCI-compatible Configuration Space. This field provides a means for a vendor to classify a particular RCRB.	RO

7.20.3. RCRB Capabilities (Offset 08h)

Figure 7-97 details allocation of register fields in the RCRB Capabilities register; Table 7-85 provides the respective bit definitions.

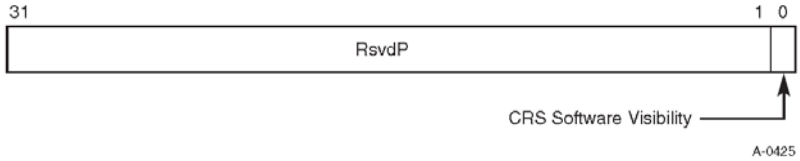


Figure 7-97: RCRB Capabilities

Table 7-85: RCRB Capabilities

Bit Location	Register Description	Attributes
0	**CRS Software Visibility** – When Set, this bit indicates that the Root Complex is capable of returning Configuration Request Retry Status (CRS) Completion Status to software for all Root Ports and integrated devices associated with this RCRB (see Section 2.3.1).	RO

7.20.4. RCRB Control (Offset 0Ch)

Figure 7-98 details allocation of register fields in the RCRB Control register; Table 7-86 provides the respective bit definitions.

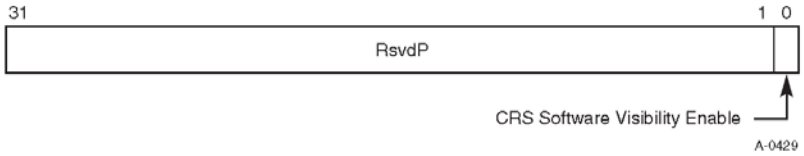


Figure 7-98: RCRB Control

Table 7-86: RCRB Control

Bit Location	Register Description	Attributes
0	**CRS Software Visibility Enable** – When Set, this bit enables the Root Complex to return Configuration Request Retry Status (CRS) Completion Status to software for all Root Ports and integrated devices associated with this RCRB (see Section 2.3.1). RCRBs that do not implement this capability must hardwire this bit to 0b. Default value of this bit is 0b.	RW

7.21. Multicast Capability

Multicast is an optional normative functionality that is controlled by the Multicast Capability structure. The Multicast Capability is applicable to Root Ports, RCRBs, Switch Ports, Endpoint Functions, and Root Complex Integrated Endpoints. It is not applicable to PCI Express to PCI/PCI-X Bridges.

In the cases of a Switch or Root Complex or a component that contains multiple Functions, multiple copies of this Capability structure are required – one for each Endpoint Function, Switch Port, or Root Port that supports Multicast. To provide implementation efficiencies, certain fields within each of the Multicast Capability structures within a component must be programmed the same and results are indeterminate if this is not the case. The fields and registers that must be configured with the same values include MC_Enable, MC_Num_Group, MC_Base_Address and MC_Index_Position. These same fields in an Endpoint's Multicast Capability structure must match those configured into a Multicast Capability structure of the Switch or Root Complex above the Endpoint or in which the Root Complex Integrated Endpoint is integrated.

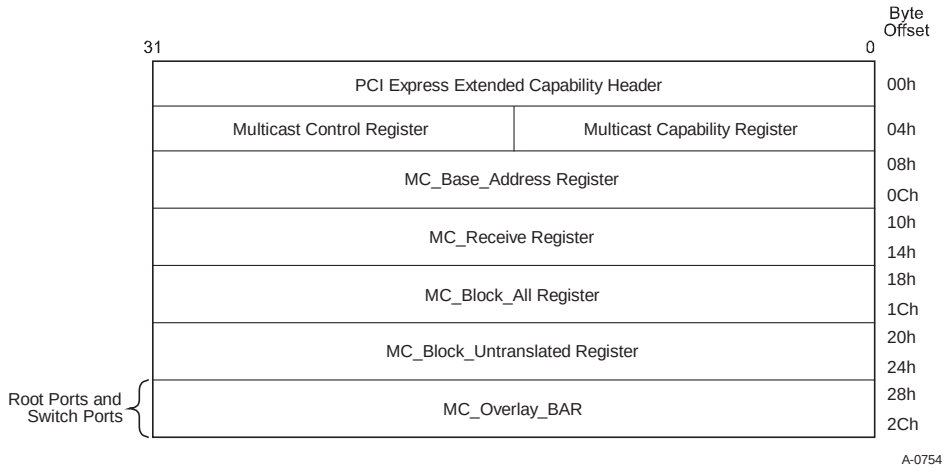


Figure 7-99: Multicast Extended Capability Structure

7.21.1. Multicast Extended Capability Header (Offset 00h)

Figure 7-100 details allocation of the fields in the Multicast Extended Capability Header and Table 7-87 provides the respective bit definitions.

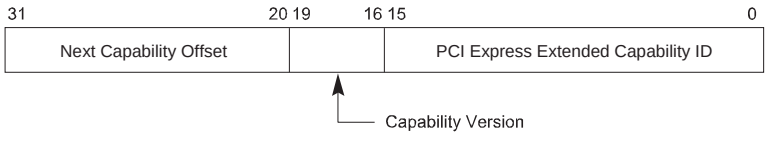


Figure 7-100: Multicast Extended Capability Header

Table 7-87: Multicast Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the Multicast Capability is 0012h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.21.2. Multicast Capability Register (Offset 04h)

Figure 7-101 details allocation of the fields in the Multicast Capability register and Table 7-88 provides the respective bit definitions.

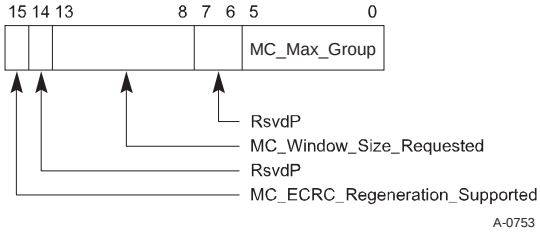


Figure 7-101: Multicast Capability Register

Table 7-88: Multicast Capability Register

Bit Location	Register Description	Attributes
5:0	**MC_Max_Group** – Value indicates the maximum number of Multicast Groups that the component supports, encoded as M-1. A value of 00h indicates that one Multicast Group is supported.	RO
13:8	**MC_Window_Size_Requested** – In Endpoints, the $\log_2$ of the Multicast Window size requested. RsvdP in Switch and Root Ports.	RO
15	**MC_ECRC_Regeneration_Supported** – If Set, indicates that ECRC regeneration is supported. This bit must not be Set unless the Function supports Advanced Error Reporting, and the ECRC Check Capable bit in the Advanced Error Capabilities and Control register is also Set. However, if ECRC regeneration is supported, its operation is not contingent upon the setting of the ECRC Check Enable bit in the Advanced Error Capabilities and Control register. This bit is applicable to Switch and Root Ports and is RsvdP in all other Functions.	RO/RsvdP

7.21.3. Multicast Control Register (Offset 06h)

Figure 7-102 details allocation of the fields in the Multicast Control register and Table 7-89 provides the respective bit definitions.

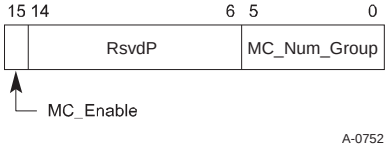


Figure 7-102: Multicast Control Register

Table 7-89: Multicast Control Register

Bit Location	Register Description	Attributes
5:0	**MC_Num_Group** – Value indicates the number of Multicast Groups configured for use, encoded as N-1. The default value of 00 0000b indicates that one Multicast Group is configured for use. Behavior is undefined if value exceeds MC_Max_Group. This parameter indirectly defines the upper limit of the Multicast address range. This field is ignored if MC_Enable is Clear. Default value is 00 0000b.	RW
15	**MC_Enable** – When Set, the Multicast mechanism is enabled for the component. Default value is 0b.	RW

7.21.4. MC_Base_Address Register (Offset 08h)

The MC_Base_Address register contains the MC_Base_Address and the MC_Index_Position. Figure 7-103 details allocation of the fields in the MC_Base_Address register and Table 7-90 provides the respective bit definitions.

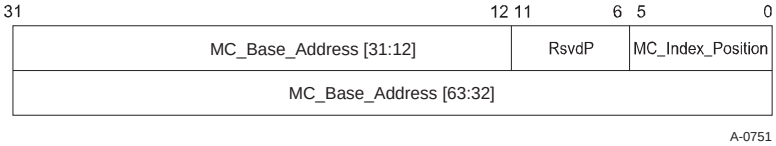


Figure 7-103: MC_Base_Address Register

Table 7-90: MC_Base Address Register

Bit Location	Register Description	Attributes
5:0	**MC_Index_Position** – The location of the LSB of the Multicast Group number within the address. Behavior is undefined if this value is less than 12 and MC_Enable is Set. Default is 0.	RW
63:12	**MC_Base_Address** – The base address of the Multicast address range. The behavior is undefined if MC_Enable is Set and bits in this field corresponding to address bits that contain the Multicast Group number or address bits less than MC_Index_Position are non-zero. Default is 0.	RW

7.21.5. MC_Receive Register (Offset 10h)

The MC_Receive register provides a bit vector denoting which Multicast groups the Function should accept, or in the case of Switch and Root Complex Ports, forward Multicast TLPs. This register is required in all Functions that implement the MC Capability structure.

Figure 7-104 details allocation of the fields in the MC_Receive register and Table 7-91 provides the respective bit definitions.

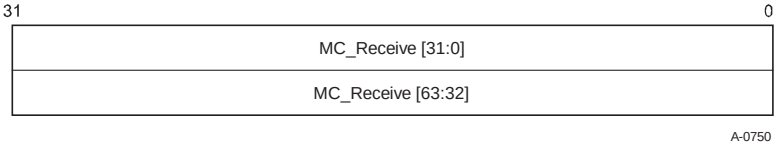


Figure 7-104: MC_Receive Register

Table 7-91: MC_Receive Register

Bit Location	Register Description	Attributes
MC_Max_Group:0	**MC_Receive** – For each bit that's Set, this Function gets a copy of any Multicast TLPs for the associated Multicast Group. Bits above MC_Num_Group are ignored by hardware. Default value of each bit is 0b.	RW
All other bits	Reserved	RsvdP

7.21.6. MC_Block_All Register (Offset 18h)

The MC_Block_All register provides a bit vector denoting which Multicast groups the Function should block. This register is required in all Functions that implement the MC Capability structure.

Figure 7-105 details allocation of the fields in the MC_Block_All register and Table 7-92 provides the respective bit definitions.

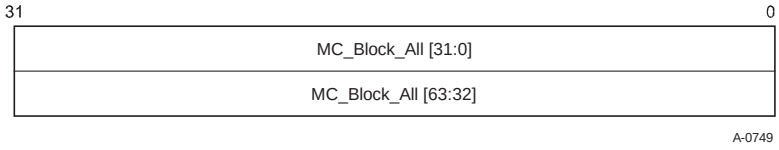


Figure 7-105: MC_Block_All Register

Table 7-92: MC_Block_All Register

Bit Location	Register Description	Attributes
MC_Max_Group:0	**MC_Block_All** – For each bit that is Set, this Function is blocked from sending TLPs to the associated Multicast Group. Bits above MC_Num_Group are ignored by hardware. Default value of each bit is 0b.	RW
All other bits	Reserved	RsvdP

7.21.7. MC_Block_Untranslated Register (Offset 20h)

The MC_Block_Untranslated register is used to determine whether or not a TLP that includes an Untranslated Address should be blocked. This register is required in all Functions that implement the MC Capability structure. However, an Endpoint Function that does not implement the ATS capability may implement this register as RsvdP.

Figure 7-106 details allocation of the fields in the MC_Block_Untranslated register and Table 7-93 provides the respective bit definitions.

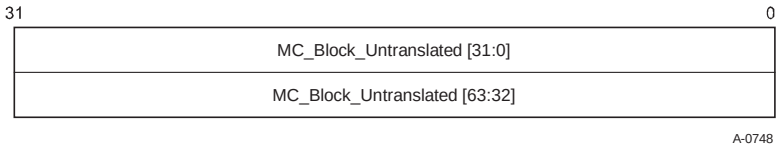


Figure 7-106: MC_Block_Untranslated Register

Table 7-93: MC_Block_Untranslated Register

Bit Location	Register Description	Attributes
MC_Max_Group:0	**MC_Block_Untranslated** – For each bit that is Set, this Function is blocked from sending TLPs containing Untranslated Addresses to the associated MCG. Bits above MC_Num_Group are ignored by hardware. Default value of each bit is 0b.	RW
All other bits	Reserved	RsvdP

7.21.8. MC_Overlay_BAR (Offset 28h)

The MC_Overlay_BAR is required in Switch and Root Complex Ports that support the Multicast Capability and not implemented in Endpoints. Software must interpret the Device/Port Type Field in the PCI Express Capabilities register to determine if the MC_Overlay_BAR is present in a Function.

The MC_Overlay_BAR specifies the base address of a window in unicast space onto which Multicast TLPs going out an Egress Port are overlaid by a process of address replacement. This allows a single BAR in an Endpoint attached to the Switch or Root Port to be used for both unicast and Multicast traffic. At a Switch Upstream Port, it allows the Multicast address range, or a portion of it, to be overlayed onto host memory.

Figure 7-107 details allocation of the fields in the MC_Overlay BAR and Table 7-94 provides the respective bit definitions.

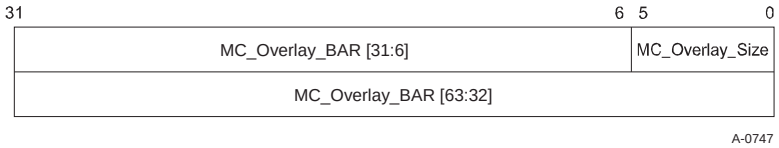


Figure 7-107: MC_Overlay_BAR

Table 7-94: MC_Overlay BAR

Bit Location	Register Description	Attributes
5:0	**MC_Overlay_Size** – If 6 or greater, specifies the size in bytes of the overlay aperture as a power of 2. If less than 6, disables the overlay mechanism. Default value is 00 0000b.	RW
63:6	**MC_Overlay_BAR** – Specifies the base address of the window onto which MC TLPs passing through this Function will be overlaid. Default value is 0.	RW

7.22. Resizable BAR Capability

The Resizable BAR Capability is an optional capability that allows hardware to communicate resource sizes, and system software, after determining the optimal size, to communicate this optimal size back to the hardware. Hardware communicates the resource sizes that are acceptable for operation via the Resizable BAR Capability register. Software determines, through a proprietary mechanism, what the optimal size is for the resource, and programs that size via the BAR Size field of the Resizable BAR Control register. Hardware immediately reflects the size inference in the read-only bits of the appropriate Base Address register. Hardware must Clear any bits that change from RW to read-only, so that subsequent reads return zero. Software must clear the Memory Space Enable bit in the Command register before writing the BAR Size field. After writing the BAR Size field, the contents of the corresponding BAR are undefined. To ensure that it contains a valid address after resizing the BAR, system software must reprogram the BAR, and Set the Memory Space Enable bit (unless the resource is not allocated).

The Resizable BAR Capability register is permitted to indicate the ability to operate at 4 GB or greater only if the associated BAR is a 64-bit BAR.

This capability is applicable to Functions that have Base Address registers only. It is strongly recommended that a Function not advertise any supported BAR sizes in its Resizable BAR Capability register that are larger than the space it would effectively utilize if allocated.

IMPLEMENTATION NOTE

Using the Capability During Resource Allocation

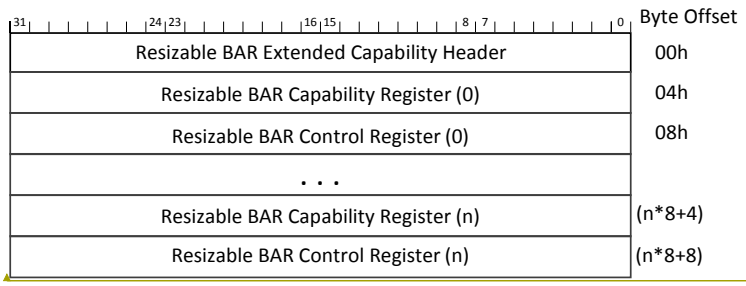
System software that allocates resources can use this capability to resize the resources inferred by the Function's BAR's read-only bits. Previous versions of this software determined the resource size by writing FFFFh to the BAR, reading back the value, and determining the size by the number of bits that are Set. Following this, the base address is written to the BAR.

System software uses this capability in place of the above mentioned method of determining the resource size, and prior to assigning the base address to the BAR. Potential usable resource sizes are reported by the Function, and read, from the Resizable BAR Capability registers. It is intended that the software allocate the largest of the reported sizes that it can, since allocating less address space than the largest reported size can result in lower performance. Software then writes the size to the Resizable BAR Command register for the appropriate BAR for the Function. Following this, the base address is written to the BAR.

For interoperability reasons, it is possible that hardware will set the default size of the BAR to a low size; that is, a size lower than the largest reported in the Resizable BAR Capability register. Software that does not use this capability to size resources will likely result in sub-optimal resource allocation, where the resources are smaller than desirable, or not allocatable because there is no room for them.

With the Resizable BAR capability, the amount of address space consumed by a device can change. In a resource constrained environment, the allocation of more address space to a device may result in allocation of less of the address space to other memory-mapped hardware, like system RAM. System software responsible for allocating resources in this kind of environment is recommended to distribute the limited address space appropriately.

The Resizable BAR Capability structure defines a PCI Express Extended Capability which is located in PCI Express Extended Configuration Space, that is, above the first 256 bytes, and is shown below in Figure 7-108. This structure allows devices with this capability to be identified and controlled. A Capability and a Control register is implemented for each BAR that is resizable. Since a maximum of six BARs may be implemented by any Function, the Resizable BAR Capability structure can range from 12 bytes long (for a single BAR) to 52 bytes long (for all six BARs).



Formatted: Font color: Black

Figure 7-108: Resizable BAR Capability

Comment [VS297]: Errata B11- widened BAR Control Registers (0) and (n)

7.22.1. Resizable BAR Extended Capability Header (Offset 00h)

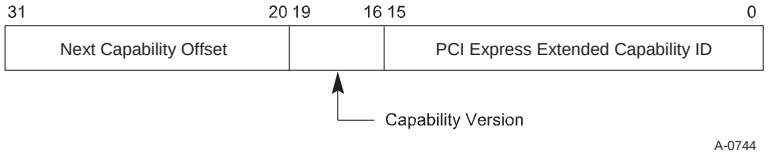


Figure 7-109: Resizable BAR Extended Capability Header

Table 7-95: Resizable BAR Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. The PCI Express Extended Capability ID for the Resizable BAR Capability is 0015h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.22.2. Resizable BAR Capability Register

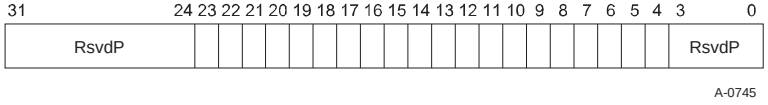


Figure 7-110: Resizable BAR Capability Register

Table 7-96: Resizable BAR Capability Register

Bit Location	Register Description	Attributes
4	When Set, indicates that the Function supports operating with the BAR sized to 1 MB	RO
5	When Set, indicates that the Function supports operating with the BAR sized to 2 MB	RO
6	When Set, indicates that the Function supports operating with the BAR sized to 4 MB	RO
7	When Set, indicates that the Function supports operating with the BAR sized to 8 MB	RO
8	When Set, indicates that the Function supports operating with the BAR sized to 16 MB	RO
9	When Set, indicates that the Function supports operating with the BAR sized to 32 MB	RO
10	When Set, indicates that the Function supports operating with the BAR sized to 64 MB	RO
11	When Set, indicates that the Function supports operating with the BAR sized to 128 MB	RO
12	When Set, indicates that the Function supports operating with the BAR sized to 256 MB	RO
13	When Set, indicates that the Function supports operating with the BAR sized to 512 MB	RO
14	When Set, indicates that the Function supports operating with the BAR sized to 1 GB	RO
15	When Set, indicates that the Function supports operating with the BAR sized to 2 GB	RO
16	When Set, indicates that the Function supports operating with the BAR sized to 4 GB	RO
17	When Set, indicates that the Function supports operating with the BAR sized to 8 GB	RO
18	When Set, indicates that the Function supports operating with the BAR sized to 16 GB	RO
19	When Set, indicates that the Function supports operating with the BAR sized to 32 GB	RO
20	When Set, indicates that the Function supports operating with the BAR sized to 64 GB	RO
21	When Set, indicates that the Function supports operating with the BAR sized to 128 GB	RO
22	When Set, indicates that the Function supports operating with the BAR sized to 256 GB	RO
23	When Set, indicates that the Function supports operating with the BAR sized to 512 GB	RO

7.22.3. Resizable BAR Control Register

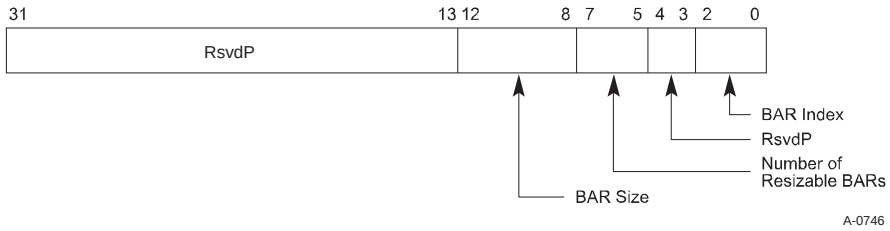


Figure 7-111: Resizable BAR Control Register

Table 7-97: Resizable BAR Control Register

Bit Location	Register Description	Attributes
2:0	**BAR Index** – This encoded value points to the beginning of the BAR. 0 = BAR located at offset 10h 1 = BAR located at offset 14h 2 = BAR located at offset 18h 3 = BAR located at offset 1Ch 4 = BAR located at offset 20h 5 = BAR located at offset 24h All other encodings are Reserved. For a 64-bit Base Address register, the BAR Index indicates the lower DWORD. This value indicates which BAR supports a negotiable size.	RO
7:5	**Number of Resizable BARs** – Indicates the total number of resizable BARs in the capability structure for the Function. See Figure 7-108. The value of this field must be in the range of 01h to 06h. The field is valid in Resizable BAR Control register (0) (at offset 008h), and is RsvdP for all others.	RO/RsvdP

Bit Location	Register Description	Attributes
12:8	**BAR Size** – This is an encoded value. 0 = 1 MB 1 = 2 MB 2 = 4 MB 3 = 8 MB … 19 = 512 GB The default value of this field is equal to the default size of the address space that the BAR resource is requesting via the BAR's read-only bits. When this register field is programmed, the value is immediately reflected in the size of the resource, as encoded in the number of read-only bits in the BAR. Software must only write supported values that correspond to those reported in the Resizable BAR Capability register. Writing an unsupported value will produce undefined results.	RW

7.23. ARI Capability

ARI is an optional capability. This capability must be implemented by each Function in an ARI Device. It is not applicable to a Root Port, a Switch Downstream Port, a Root Complex Integrated Endpoint, or a Root Complex Event Collector.

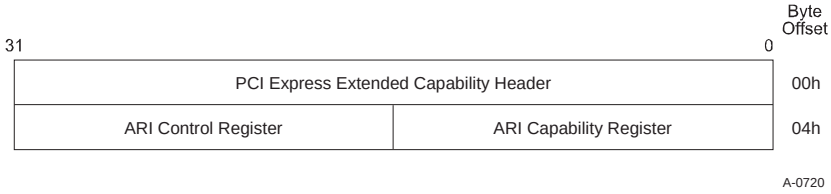


Figure 7-112: ARI Capability

7.23.1. ARI Capability Header (Offset 00h)

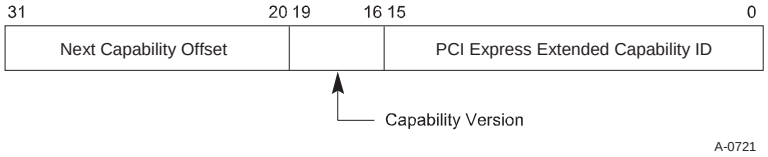


Figure 7-113: ARI Capability Header

Table 7-98: ARI Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. PCI Express Extended Capability ID for the ARI Capability is 000Eh.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.23.2. ARI Capability Register (Offset 04h)

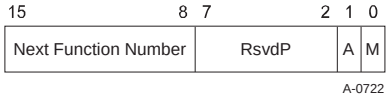


Figure 7-114: ARI Capability Register

Table 7-99: ARI Capability Register

Bit Location	Register Description	Attributes
0	**MFVC Function Groups Capability (M)** – Applicable only for Function 0; must be 0b for all other Functions. If 1b, indicates that the ARI Device supports Function Group level arbitration via its Multi-Function Virtual Channel (MFVC) Capability structure.	RO
1	**ACS Function Groups Capability (A)** – Applicable only for Function 0; must be 0b for all other Functions. If 1b, indicates that the ARI Device supports Function Group level granularity for ACS P2P Egress Control via its ACS Capability structures.	RO

Bit Location	Register Description	Attributes
15:8	**Next Function Number** – This field indicates the Function Number of the next higher numbered Function in the Device, or 00h if there are no higher numbered Functions. Function 0 starts this linked list of Functions.	RO

7.23.3. ARI Control Register (Offset 06h)

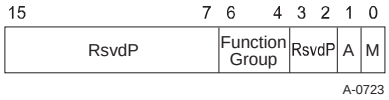


Figure 7-115: ARI Control Register

Table 7-100: ARI Control Register

Bit Location	Register Description	Attributes
0	**MFVC Function Groups Enable (M)** – Applicable only for Function 0; must be hardwired to 0b for all other Functions. When set, the ARI Device must interpret entries in its Function Arbitration Table as Function Group Numbers rather than Function Numbers. Default value of this bit is 0b. Must be hardwired to 0b if the MFVC Function Groups Capability bit is 0b.	RW
1	**ACS Function Groups Enable (A)** – Applicable only for Function 0; must be hardwired to 0b for all other Functions. When set, each Function in the ARI Device must associate bits within its Egress Control Vector with Function Group Numbers rather than Function Numbers. Default value of this bit is 0b. Must be hardwired to 0b if the ACS Function Groups Capability bit is 0b.	RW
6:4	**Function Group** – Assigns a Function Group Number to this Function. Default value of this field is 000b. Must be hardwired to 000b if in Function 0, the MFVC Function Groups Capability bit and ACS Function Groups Capability bit are both 0b.	RW

7.24. Dynamic Power Allocation (DPA) Capability

The DPA Capability structure is shown in Figure 7-116.

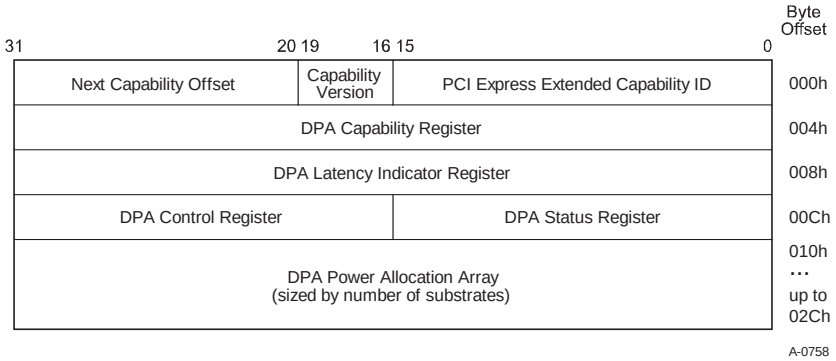


Figure 7-116: Dynamic Power Allocation Capability Structure

7.24.1. DPA Extended Capability Header (Offset 00h)

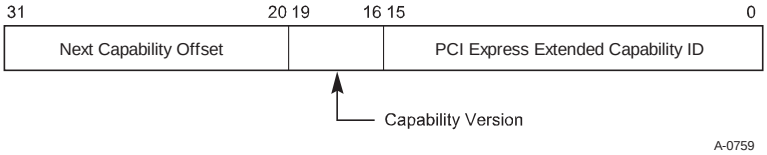


Figure 7-117: DPA Extended Capability Header

Table 7-101: DPA Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the DPA Extended Capability is 0016h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.24.2. DPA Capability Register (Offset 04h)

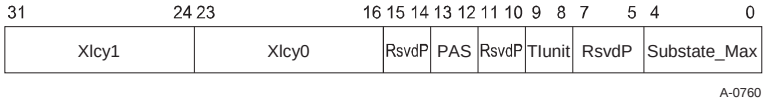


Figure 7-118: DPA Capability Register

Table 7-102: DPA Capability Register

Bit Location	Register Description	Attributes
4:0	**Substate_Max** – Value indicates the maximum substate number, which is the total number of supported substates minus one. A value of 0 0000b indicates support for one substate.	RO
9:8	**Transition Latency Unit (Tlunit)** – A substate's Transition Latency Value is multiplied by the Transition Latency Unit to determine the maximum Transition Latency for the substate. Defined encodings are 00b – 1 ms 01b – 10 ms 10b – 100 ms 11b – Reserved	RO
13:12	**Power Allocation Scale (PAS)** – The encodings provide the scale to determine power allocation per substate in Watts. The value corresponding to the substate in the Substate Power Allocation field is multiplied by this field to determine the power allocation for the substate. Defined encodings are 00b – 10.0x 01b – 1.0x 10b – 0.1x 11b – 0.01x	RO
23:16	**Transition Latency Value 0 (Xlcy0)** – This value is multiplied by the Transition Latency Unit to determine the maximum Transition Latency for the substate	RO
31:24	**Transition Latency Value 1 (Xlcy1)** – This value is multiplied by the Transition Latency Unit to determine the maximum Transition Latency for the substate.	RO

7.24.3. DPA Latency Indicator Register (Offset 08h)

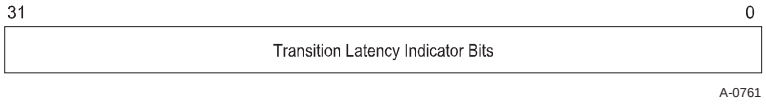


Figure 7-119: DPA Latency Indicator Register

Table 7-103: DPA Latency Indicator Register

Bit Location	Register Description	Attributes
Substate_Max:0	**Transition Latency Indicator Bits** – Each bit indicates which Transition Latency Value is associated with the corresponding substate. A value of 0b indicates Transition Latency Value 0; a value of 1b indicates Transition Latency Value 1.	RO
All other bits	Reserved	RsvdP

7.24.4. DPA Status Register (Offset 0Ch)

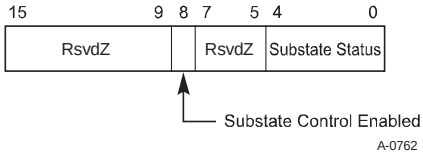


Figure 7-120: DPA Status Register

Table 7-104: DPA Status Register

Bit Location	Register Description	Attributes
4:0	**Substate Status** – Indicates current substate for this Function. Default is 0 0000b.	RO
8	**Substate Control Enabled** – Used by software to disable the Substate Control field in the DPA Control register. Hardware sets this bit following a Conventional Reset or FLR. Software clears this bit by writing a 1b to it. Software is unable to set this bit directly. When this bit is Set, the Substate Control field determines the current substate. When this bit is Clear, the Substate Control field has no effect on the current substate. Default value is 1b.	RW1C

7.24.5. DPA Control Register (Offset 0Eh)



Figure 7-121: DPA Control Register

Table 7-105: DPA Control Register

Bit Location	Register Description	Attributes
4:0	**Substate Control** – Used by software to configure the Function substate. Software writes the substate value in this field to initiate a substate transition. When the Substate Control Enabled bit in the DPA Status register is Set, this field determines the Function substate. When the Substate Control Enabled bit in the DPA Status register is Clear, this field has no effect on the Function substate. Default value is 0 0000b.	RW

7.24.6. DPA Power Allocation Array

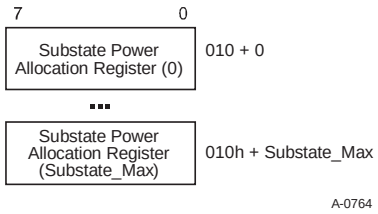


Figure 7-122: DPA Power Allocation Array

Each Substate Power Allocation register indicates the power allocation value for its associated substate. The number of Substate Power Allocation registers implemented must be equal to the number of substates supported by Function, which is Substate_Max plus one.

Table 7-106: Substate Power Allocation Register (0 to Substate_Max)

Bit Location	Register Description	Attributes
7:0	**Substate Power Allocation** – The value in this field is multiplied by the Power Allocation Scale to determine power allocation in Watts for the associated substate.	RO

7.25. Latency Tolerance Reporting (LTR) Capability

The PCI Express Latency Tolerance Reporting (LTR) Capability is an optional Extended Capability that allows software to provide platform latency information to components with Upstream Ports (Endpoints and Switches), and is required for Switch Upstream Ports and Endpoints if the Function supports the LTR mechanism. It is not applicable to Root Ports, Bridges, or Switch Downstream Ports.

For a multi-Function device associated with the Upstream Port of a component that implements the LTR mechanism, this Capability structure must be implemented only in Function 0, and must control the component's Link behavior on behalf of all the Functions of the device.

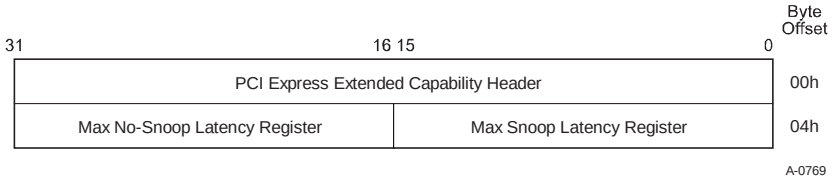


Figure 7-123: LTR Extended Capability Structure

7.25.1. LTR Extended Capability Header (Offset 00h)

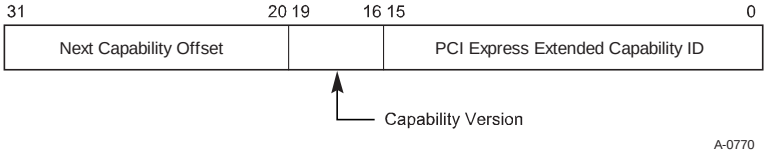


Figure 7-124: LTR Extended Capability Header

Table 7-107: LTR Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability for the LTR Extended Capability is 0018h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.25.2. Max Snoop Latency Register (Offset 04h)

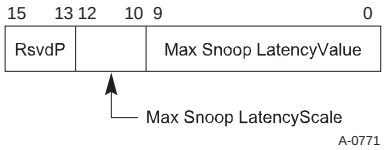


Figure 7-125: Max Snoop Latency Register

Table 7-108: Max Snoop Latency Register

Bit Location	Register Description	Attributes
9:0	**Max Snoop LatencyValue** – Along with the Max Snoop LatencyScale field, this register specifies the maximum snoop latency that a device is permitted to request. Software should set this to the platform's maximum supported latency or less. The default value for this field is 00 0000 0000b.	RW
12:10	**Max Snoop LatencyScale** – This register provides a scale for the value contained within the Maximum Snoop LatencyValue field. Encoding is the same as the LatencyScale fields in the LTR Message. See Section 6.18. The default value for this field is 000b. Hardware operation is undefined if software writes a Not Permitted value to this field.	RW

7.25.3. Max No-Snoop Latency Register (Offset 06h)

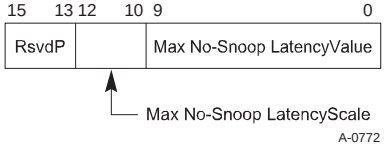


Figure 7-126: Max No-Snoop Latency Register

Table 7-109: Max No-Snoop Latency Register

Bit Location	Register Description	Attributes
9:0	**Max No-Snoop LatencyValue** – Along with the Max No-Snoop LatencyScale field, this register specifies the maximum no-snoop latency that a device is permitted to request. Software should set this to the platform's maximum supported latency or less. The default value for this field is 00 0000 0000b.	RW
12:10	**Max No-Snoop LatencyScale** – This register provides a scale for the value contained within the Max No-Snoop LatencyValue field. Encoding is the same as the LatencyScale fields in the LTR Message. See Section 6.18. The default value for this field is 000b. Hardware operation is undefined if software writes a Not Permitted value to this field.	RW

7.26. TPH Requester Capability

The TPH Requester Capability structure is required for all Functions that are capable of generating Request TLPs with TPH. For a multi-Function device, this capability must be present in each Function that is capable of generating Request TLPs with TPH.

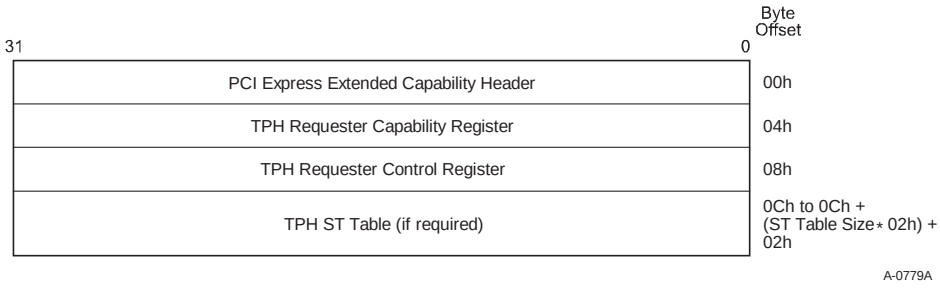


Figure 7-127: TPH Extended Capability Structure

7.26.1. TPH Requester Extended Capability Header (Offset 00h)

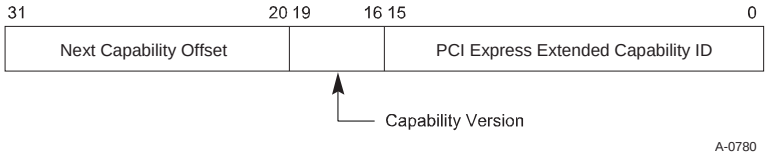


Figure 7-128: TPH Requester Extended Capability Header

Table 7-110: TPH Requester Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the TPH Requester Capability is 0017h**.**	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.26.2. TPH Requester Capability Register (Offset 04h)

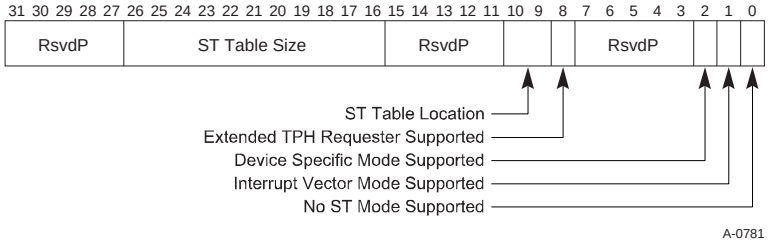


Figure 7-129: TPH Requester Capability Register

Table 7-111: TPH Requester Capability Register

Bit Location	Register Description	Attributes
0	**No ST Mode Supported** – If set indicates that the Function supports the No ST Mode of operation. This mode is required to be supported by all Functions that implement this Capability structure. This bit must have a value of 1b.	RO
1	**Interrupt Vector Mode Supported** – If set indicates that the Function supports the Interrupt Vector Mode of operation.	RO
2	**Device Specific Mode Supported** – If set indicates that the Function supports the Device Specific Mode of operation.	RO
8	**Extended TPH Requester Supported** – If Set indicates that the Function is capable of generating Requests with a TPH TLP Prefix. See Section 2.2.7.1 for additional details.	RO
10:9	**ST Table Location** – Value indicates if and where the ST Table is located. Defined Encodings are: 00b – ST Table is not present. 01b – ST Table is located in the TPH Requester Capability structure. 10b – ST Table is located in the MSI-X Table (see Section 7.7). 11b – Reserved A Function that only supports the No ST Mode of operation must have a value of 00b in this field. A Function may report a value of 10b only if it implements an MSI-X Capability.	RO
26:16	**ST Table Size** – Value indicates the maximum number of ST Table entries the Function may use. Software reads this field to determine the ST Table Size N, which is encoded as N-1. For example, a returned value of 000 0000 0011b indicates a table size of four entries. There is an upper limit of 64 entries when the ST Table is located in the TPH Requester Capability structure. ~~There is an upper limit of 2048 entries w~~When the ST Table is located in the MSI-X Table, this value is limited by the size of the MSI-X Table. This field is only applicable for Functions that implement an ST Table as indicated by the ST Table Location field. Otherwise, the value in this field is undefined.	RO

Comment [VS298]: Errata B6

Comment [VS299]: Errata A8

7.26.3. TPH Requester Control Register (Offset 08h)

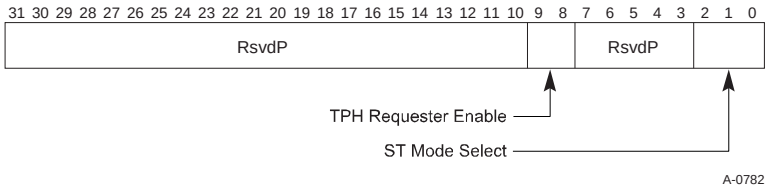


Figure 7-130: TPH Requester Control Register

Table 7-112: TPH Requester Control Register

Bit Location	Register Description	Attributes
2:0	**ST Mode Select** – selects the ST Mode of operation. Defined encodings are: 000b – No ST Mode 001b – Interrupt Vector Mode 010b – Device Specific Mode all other encodings – reserved for future use Functions that support only the No ST Mode of operation must hardwire this field to 000b. Function operation is undefined if software enables a mode of operation that does not correspond to a mode supported by the Function. The default value of this field is 000b. See Section 6.17.3 for details on ST modes of operation.	RW
9:8	**TPH Requester Enable** – Controls the ability to issue Request TLPs using either TPH or Extended TPH. Defined encodings are: 00b – Function operating as a Requester is not permitted to issue Requests with TPH or Extended TPH. 01b – Function operating as a Requester is permitted to issue Requests with TPH and is not permitted to issue Requests with Extended TPH. 10b – Reserved. 11b – Function operating as a Requester is permitted to issue Requests with TPH and Extended TPH. Functions that advertise that they do not support Extended TPH are permitted to hardwire bit 9 of this field to 0b. The default value of this field is 00b.	RW

Comment [VS300]: Errata B33 (Part C list resolved 10/13)

7.26.4. TPH ST Table (Starting from Offset 0Ch)

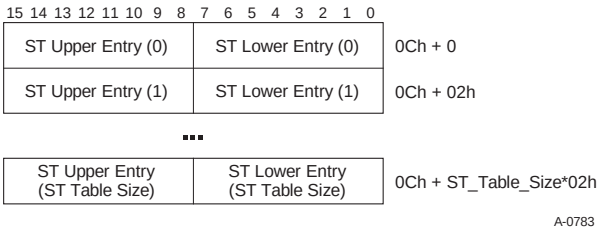


Figure 7-131: TPH ST Table

The ST Table must be implemented in the TPH Requester Capability structure if the value of the ST Table Location field is 01b. For all other values, the ST Entry registers must not be implemented. Each implemented ST Entry is 16 bits. The number of ST Entry registers implemented must be equal to the number of ST Table entries supported by the Function, which is the value of the ST Table Size field plus one.

Table 7-113: TPH ST Table

Bit Location	Register Description	Attributes
7:0	**ST Lower** – This field contains the lower 8 bits of a Steering Tag. Default value of this field is 00h.	RW
15:8	**ST Upper** – If the Function's Extended TPH Requester Supported bit is Set, then this field contains the upper 8 bits of a Steering Tag. Otherwise, this field is RsvdP. Default value of this field is 00h.	RW

7.27. Secondary PCI Express Extended Capability

The Secondary PCI Express Extended Capability structure must be implemented in any Function or RCRB where any of the following are true:

- The Supported Link Speeds Vector field indicates that the Link supports Link Speeds of 8.0 GT/s or higher (see Section 7.8.18 or 7.14.2).
- Any bit in the Lower SKP OS Generation Supported Speeds Vector field is Set (see Section 7.8.18).
- When Lane based errors are reported in the Lane Error Status register (see Section 4.2.6).

To support future additions to this capability, this capability ~~may be present~~ is permitted in any Function or RCRB associated with a Link. For a Multi-Function Device associated with an Upstream Port, this capability is permitted only in Function 0 of the Device.

Comment [VS301]: Errata B29 and SRIS ECN; resolved by Errata Part C list (Octber 2013).

Comment [VS302]: "is permitted" per: Errata Part C list resolved 10/13

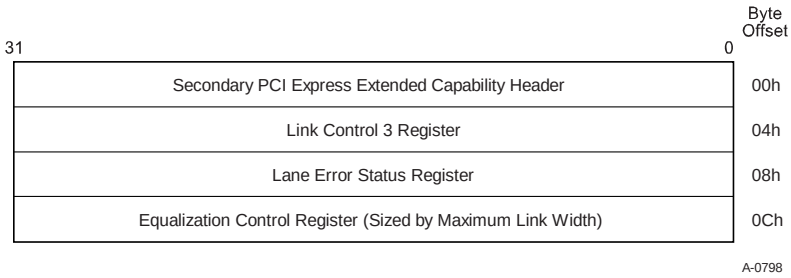


Figure 7-132: Secondary PCI Express Extended Capability Structure

7.27.1. Secondary PCI Express Extended Capability Header (Offset 00h)

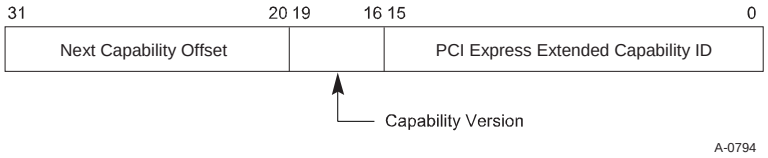


Figure 7-133: Secondary PCI Express Extended Capability Header

Table 7-114: Secondary PCI Express Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the Secondary PCI Express Extended Capability is 0019h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	RO

7.27.2. Link Control 3 Register (Offset 04h)

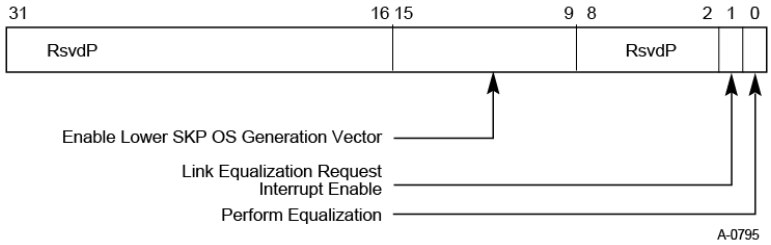


Figure 7-134: Link Control 3 Register

Table 7-115: Link Control 3 Register

Bit Location	Register Description	Attributes
0	**Perform Equalization** – When this bit is 1b and a 1b is written to the Retrain Link bit with the Target Link Speed field set to 8.0 GT/s, the Downstream Port must perform Link Equalization. Refer to Sections 4.2.3 and 4.2.6.4.2 for details. This bit is RW for Downstream Ports and for Upstream Ports when Crosslink Supported is 1b (see Section 7.8.18). This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. The default value is 0b. If the Port does not support 8.0 GT/s, this bit is permitted to be hardwired to 0b.	RW/RsvdP

Comment [VS303]: Errata B27

Bit Location	Register Description	Attributes
1	**Link Equalization Request Interrupt Enable** – When Set, this bit enables the generation of an interrupt to indicate that the Link Equalization Request bit has been set. This bit is RW for Downstream Ports and for Upstream Ports when Crosslink Supported is 1b (see Section 7.8.18). This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. The default value for this bit is 0b. If the Port does not support 8.0 GT/s, this bit is permitted to be hardwired to 0b.	RW/RsvdP
9:15	**Enable Lower SKP OS Generation Vector** – When the Link is in L0 and the bit in this field corresponding to the current Link speed is Set, SKP Ordered Sets are scheduled at the rate defined for SRNS, overriding the rate required based on the clock tolerance architecture. See Section 4.2.7 for additional requirements. Bit definitions within this field are: Bit 0 2.5 GT/s Bit 1 5.0 GT/s Bit 2 8.0 GT/s Bits 6:3 RsvdP Bits in this field are RW if the corresponding bit in the Lower SKP OS Generation Supported Speeds Vector is Set, otherwise they are permitted to be hardwired to 0. Behavior is undefined if a bit is Set in this field and the corresponding bit in the Lower SKP OS Generation Supported Speeds Vector is not Set. The default value of this field is 000 0000b.	RW/RsvdP

Comment [VS304]: Errata B27

Comment [VS305]: ECN:SRIS

7.27.3. Lane Error Status Register (Offset 08h)

The Lane Error Status register consists of a 32-bit vector, where each bit indicates if the Lane with the corresponding Lane number detected an error. This Lane number is the default Lane number which is invariant to Link width and Lane reversal negotiation that occurs during Link training.

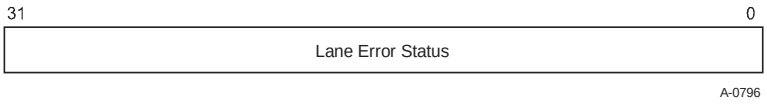


Figure 7-135: Lane Error Status Register

Table 7-116: Lane Error Status Register

Bit Location	Register Description	Attributes
(Maximum Link Width -1):0	**Lane Error Status Bits** – Each bit indicates if the corresponding Lane detected a Lane-based error. A value of 1b indicates that a Lane based-error was detected on the corresponding Lane Number (see Sections 4.2.2.3.3, 4.2.6, and 4.2.7.2 for details). The default value of each bit is 0b. For ~~Links~~ Ports that are narrower than 32 ~~bits~~Lanes, the unused upper bits [31:~~Bits~~ Maximum Link Width] are RsvdZ. For Ports that do not support 8.0 GT/s and do not set these bits based on 8b/10b errors (optional, see Section 4.2.6), this field is permitted to be hardwired to 0.	RW1CS

Comment [VS306]: Errata B27

7.27.4. Lane Equalization Control Register (Offset 0Ch)

The Equalization Control register consists of control fields required for per Lane equalization and the number of entries in this register are sized by Maximum Link Width (see Section 7.8.6). Each entry contains the values for the Lane with the corresponding default Lane number which is invariant to Link width and Lane reversal negotiation that occurs during Link training.

If the Port does not support 8.0 GT/s, this register is permitted to be hardwired to 0.

Comment [VS307]: Errata B27

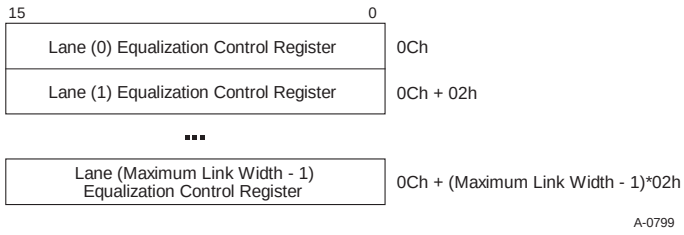


Figure 7-136: Lane Equalization Control Register

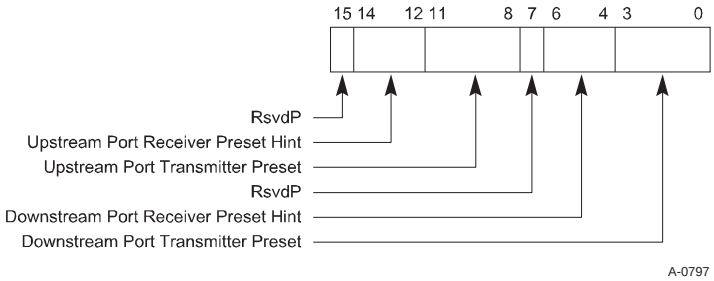


Figure 7-137: Lane ((Maximum Link Width – 1):0) Equalization Control Register

Table 7-117: Lane ((Maximum Link Width – 1):0) Equalization Control Register

Bit Location	Register Description	Attributes
3:0	**Downstream Port Transmitter Preset** – Transmitter Preset used for equalization by this Port when the Port is operating as a Downstream Port. This field is ignored when the Port is operating as an Upstream Port. See Section 4.3 for details. The field encodings are defined in Section 4.2.3.2. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP. Otherwise, this field is HwInit. See Section 7.8.18. The default value is 1111b.	HwInit/RsvdP (see description)
6:4	**Downstream Port Receiver Preset Hint** – Receiver Preset Hint may be used as a hint for receiver equalization by this Port when the Port is operating as a Downstream Port. This field is ignored when the Port is operating as an Upstream Port. See Section 4.3 for details. The field encodings are defined in Section 4.2.3.2. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP. Otherwise, this field is HwInit. See Section 7.8.18. The default value is 111b.	HwInit/RsvdP (see description)

<table>
<tr><th>Bit Location</th><th>Register Description</th><th>Attributes</th></tr>
<tr><td>11:8</td><td>Upstream Port Transmitter Preset – Field contains the Transmit Preset value sent or received during Link Equalization. Field usage varies as follows:
<table>
<tr><th></th><th>Operating Port Direction</th><th>Crosslink Supported</th><th>Usage</th></tr>
<tr><td>A</td><td>Downstream Port</td><td>Any</td><td>Field contains the value sent on the associated Lane during Link Equalization.

Field is HwInit.</td></tr>
<tr><td>B</td><td>Upstream Port</td><td>0b</td><td>Field is intended for debug and diagnostics. It contains the value captured from the associated Lane during Link Equalization.

Field is RO.

Note: When crosslinks are supported, case C (below) applies and this captured information is not visible to software. Vendors are encouraged to provide an alternate mechanism to obtain this information.</td></tr>
<tr><td>C</td><td>Upstream Port</td><td>1b</td><td>Field is not used or affected by the current Link Equalization.

Field value will be used if a future crosslink negotiation switches the Operating Port Direction so that case A (above) applies.

Field is HwInit.</td></tr>
</table>
See Sections 4.2.3 and 4.3 for details. The field encodings are defined in Section 4.2.3.2.

The default value is 1111b.</td><td>HwInit/RO (see description)</td></tr>
</table>

<table>
<tr><th>Bit Location</th><th>Register Description</th><th>Attributes</th></tr>
<tr><td>14:12</td><td>Upstream Port Receiver Preset Hint – Field contains the Receiver Preset Hint value sent or received during Link Equalization. Field usage varies as follows:
<table>
<tr><th></th><th>Operating Port Direction</th><th>Crosslink Supported</th><th>Usage</th></tr>
<tr><td>A</td><td>Downstream Port</td><td>Any</td><td>Field contains the value sent on the associated Lane during Link Equalization.

Field is HwInit.</td></tr>
<tr><td>B</td><td>Upstream Port</td><td>0b</td><td>Field is intended for debug and diagnostics. It contains the value captured from the associated Lane during Link Equalization.

Field is RO.

Note: When crosslinks are supported, case C (below) applies and this captured information is not visible to software. Vendors are encouraged to provide an alternate mechanism to obtain this information.</td></tr>
<tr><td>C</td><td>Upstream Port</td><td>1b</td><td>Field is not used or affected by the current Link Equalization.

Field value will be used if a future crosslink negotiation switches the Operating Port Direction so that case A (above) applies.

Field is HwInit.</td></tr>
</table>
See Sections 4.2.3 and 4.3 for details. The field encodings are defined in Section 4.2.3.2.

The default value is 111b.</td><td>HwInit/RO (see description)</td></tr>
</table>

7.28. M-PCIe Extended Capability

The M-PCIe Capability is an optional Extended Capability that is required for all ports and RCRBs that support M-PCIe. It is not applicable to Root Complex Integrated Endpoints, or Root Complex Event Collectors. The M-PCIe Extended Capability is shown in Figure 7-138.

For a multi-Function device associated with the Upstream Port of a component that implements M-PCIe, this Extended Capability Structure must be implemented only in Function 0, and must control the LINK behavior on behalf of all the Functions of the device.

This Capability must be implemented when the Port is ~~operating with~~using an M-PHY based ~~p~~Physical ~~l~~Layer ~~as defined in chapter 8~~. This C~~e~~apability must not be implemented when the Port is ~~operating with~~using a PCIe ~~PHY~~Physical Layer ~~as defined in chapter 4~~.

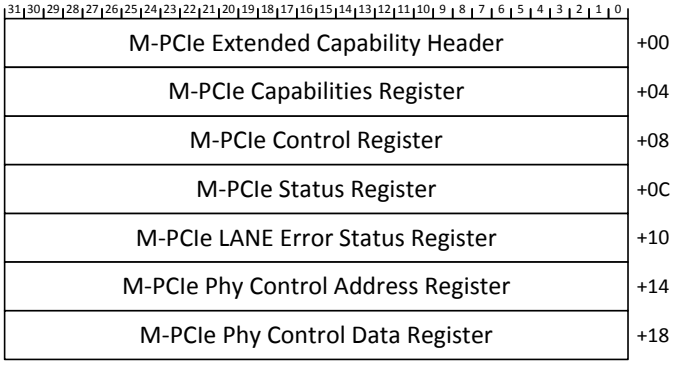


Figure 7-138. M-PCIe Extended Capability

7.28.1. M-PCIe Extended Capability Header (Offset 00h)

Figure 7-139. details allocation of register fields in the M-PCIe Extended Capability header; Table 7-118 provides the respective bit definitions.

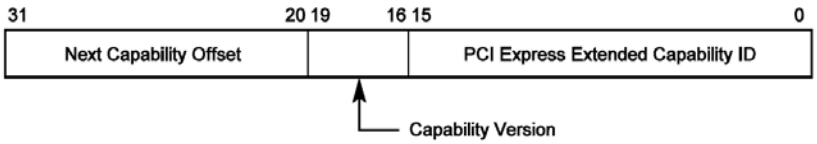


Figure 7-139. M-PCIe Extended Capability Header

Comment [VS308]: Entire section 7.28 added per ECN: M-PCIe

Comment [VS309]: Mpcie_28August2014 Errata C7

Field Code Changed

Table 7-118: M-PCIe Extended Capability Header

Bit Location	Register Description	Attributes
15:0	PCI Express Extended Capability ID – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for M-PCIe is 0020h.	RO
19:16	Capability Version – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	Next Capability Offset – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits.	RO

7.28.2. M-PCIe Capabilities Register (Offset 04h)

Figure 7-140 details allocation of register fields in the M-PCIe Capabilities register; Table 7-119 provides the respective bit definitions.

Field Code Changed

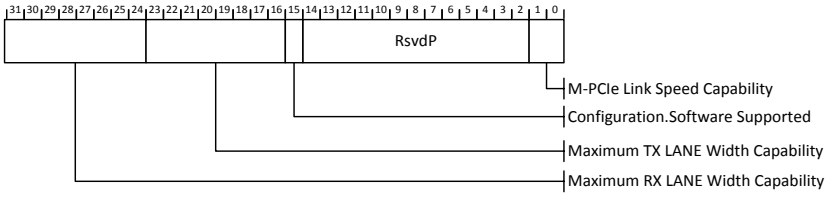


Figure 7-140: M-PCIe Capabilities Register

Table 7-119: M-PCIe Capabilities Register

Bit Location	Register Description	Attributes
1:0	M-PCIe Link Speed Capability – This Field is used to advertise the support for M-PHY HS-GEARs The encoding is as follows: Bit 0 – When set indicates that HS-G1 is supported. Bit 1 - When set indicates that HS-G2 is supported.	Hwinit
14:2	Reserved	RsvdP
15	Configuration.Software Supported – This field is used to advertise the support for Configuration.Software state. When set indicates that the Downstream Port supports Configuration.Software state. When clear indicates that the Downstream Port does not support Configuration.Software state. This bit is hardwired to 0b for Upstream Ports. See Section 8.4.9.3.2 for details.	Hwinit
23:16	Maximum TX LANE Width Capability – This field reflects the value from the TX-LANE_WIDTH_SUPPORTED field in M-PCIe Capability Attributes.	Hwinit
31:24	Maximum RX LANE Width Capability – This field reflects the value from the RX-LANE_WIDTH_SUPPORTED field in M-PCIe Capability Attributes.	Hwinit

7.28.3. M-PCIe Control Register (Offset 08h)

Figure 7-141 details allocation of register fields in the M-PCIe Control register; Table 7-120 provides the respective bit definitions.

Field Code Changed

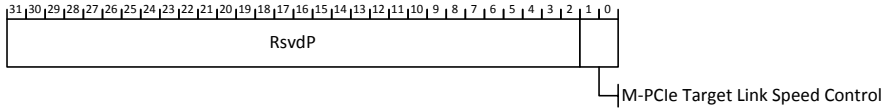


Figure 7-141: M-PCIe Control Register

Table 7-120: M-PCIe Control Register

Bit Location	Register Description	Attributes
1:0	M-PCIe Target Link Speed Control – This field is used to set the target LINK Speed to HS-G1 or HS-G2. This field is only applicable if the Target Link Speed in Link Control 2 Register has a value of 0001b. This field has no effect for all other values of Target Link Speed Control. The encoding is as follows: 01b – Indicates that HS-G1 is selected 10b - Indicates that HS-G2 is selected. All other encodings are Reserved. LINK behavior is undefined if a Reserved encoding is used. Default value of this field is 01b.	RW
31:2	Reserved	RsvdP

7.28.4. M-PCIe Status Register (Offset 0Ch)

Figure 7-142 details allocation of register fields in the M-PCIe Status register; Table 7-121 provides the respective bit definitions.

Field Code Changed

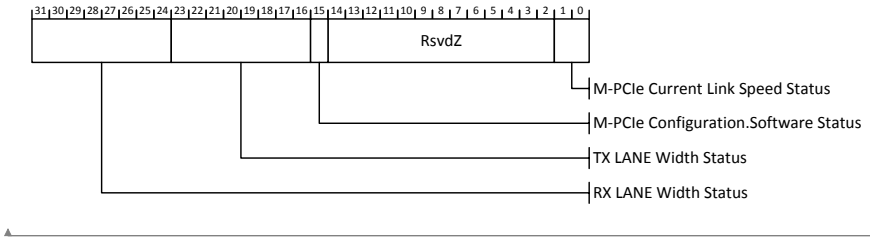


Figure 7-142: M-PCIe Status Register

Table 7-121: M-PCIe Status Register

Bit Location	Register Description	Attributes
1:0	M-PCIe Current Link Speed Status – This field indicates the current LINK Speed when the Current Link Speed field in the Link Status Register has a value of 0001b. This field is not applicable for all other values of Current Link Speed Register. The encoding is as follows: Bit 0 – When set indicates that HS-G1 is selected, when clear indicates that HS-G1 is not selected. Bit 1 - When set indicates that HS-G2 is selected, when clear indicates that HS-G2 is not selected. Only one of the two speeds are permitted to be set. Default value of this field is 01b.	RO
14:2	Reserved	RsvdZ
15	M-PCIe Configuration.Software Status – If Set, indicates that the LTSSM is in the Configuration.Software state. If Clear indicates that the LTSSM is not in the Configuration.Software state. This bit is RsvdZ for Upstream Ports.	RO/RsvdZ

Bit Location	Register Description	Attributes
23:16	TX LANE Width Status – This field reflects the current TX-LANE width. 0000 0001b – x1 TX-LANE 0000 0010b – x2 TX-LANE 0000 0100b – x4 TX-LANE 0000 1000b – x8 TX-LANE 0000 1100b – x12 TX-LANE 0001 0000b – x16 TX-LANE 0010 0000b – x32 TX-LANE All other encodings are reserved.	RO
31:24	RX LANE Width Status – This field reflects the current RX-LANE width. 0000 0001b – x1 RX-LANE 0000 0010b – x2 RX-LANE 0000 0100b – x4 RX-LANE 0000 1000b – x8 RX-LANE 0000 1100b – x12 RX-LANE 0001 0000b – x16 RX-LANE 0010 0000b – x32 RX-LANE All other encodings are reserved.	RO

7.28.5. M-PCIe LANE Error Status Register (Offset 10h)

The M-PCIe Lane Error Status register consists of a 32-bit vector, where each bit indicates if the LANE with the corresponding LANE number detected an error.

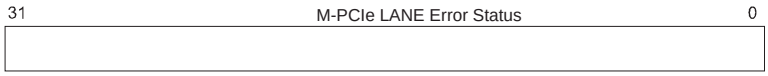


Figure 7-143: M-PCIe LANE Error Status Register

Table 7-122: M-PCIe LANE Error Status Register

Bit Location	Register Description	Attributes
(Maximum Link Width -1):0	M-PCIe LANE Error Status Bits – Each bit indicates if the corresponding LANE detected a Lane-based error. A value of 1b indicates that a LANE based-error was detected on the corresponding LANE Number (see Section 8.5 for details). The default value of each bit is 0b. For Links that are narrower than 32 bits, the unused upper bits [31:Bits Maximum Link Width] are RsvdZ	RW1CS

7.28.6. M-PCIe Phy Control Address Register (Offset 14h)

This register is required for Downstream Ports when the Configuration.Software Supported bit is Set in the M-PCIe Capabilities register. Software must not use this register when the Configuration.Software Supported bit is Clear in the M-PCIe Capabilities register. This register is not supported and all fields are RsvdP for Upstream Ports.

Figure 7-144 details allocation of register fields in the M-PCIe Phy Control Address Register; Table 7-123 provides the respective bit definitions.

Field Code Changed

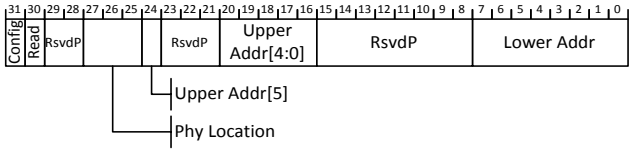


Figure 7-144: M-PCIe Phy Control Address Register

Table 7-123: M-PCIe Phy Control Address Register

Bit Location	Register Description	Attributes
7:0	Lower Addr[7:0] – This field corresponds to Lower Addr[7:0] in an RRAP packet. See Table 8-6 for details. Default is 00h.	RW
15:8	Reserved	RsvdP
20:16	Upper Addr[4:0] – Corresponds to Upper Addr[4:0] in an RRAP packet. See Table 8-6 for details. Default is 00000b.	RW
23:21	Reserved	RsvdP
24	Upper Addr[5] – Corresponds to Upper Addr[5] in an RRAP packet. See Table 8-6 for details. Default is 0b.	RW
27:25	Phy Location – Indicates the location along the Link of the Phy. The following encodings are defined: 000 Local Phy – the Phy associated with this Port 001 Remote Phy – the Phy associated with the Port on the other end of this Link Default is 000b.	RW
29:28	Reserved	RsvdP
30	Read – If Set, initiates a read operation of the register described by Phy Location, UpperAddr and LowerAddr. If clear, no read operation is being initiated. The effect of a write with this bit Set when Phy Control Pending is also Set is undefined. This bit always reads as 0b.	RW

Bit Location	Register Description	Attributes
31	Config – If Set, the Downstream Port LTSSM will stay in the Configuration.Software state. If Clear, the Downstream Port LTSSM will exit the Configuration.Software state. See Section 8.4.9.3 for details. ~~Default is 0b.~~ Default is implementation specific.	RW

Comment [VS310]: Mpcie_28August2014 Errata B11

7.28.7. M-PCIe Phy Control Data Register (Offset 18h)

This register is required for Downstream Ports when the Configuration.Software Supported bit is Set in the M-PCIe Capabilities register. Software must not use this register when the Configuration.Software Supported bit is Clear in the M-PCIe Capabilities register. This register is not supported and all fields are RsvdP for Upstream Ports.

Figure 7-145 details allocation of register fields in the M-PCIe Phy Control Data Register; Table 7-124: provides the respective bit definitions.

Field Code Changed

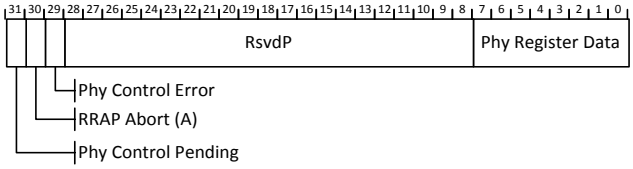


Figure 7-145: M-PCIe Phy Control Data Register

Table 7-124: M-PCIe Phy Control Data Register

Bit Location	Register Description	Attributes
7:0	**Phy Register Data** – When this register is written, a Phy Control Write request is issued to the Local Phy (Phy Location 0) or the Remote Phy (Phy Location 1). When read, this field returns data associated with the most recent Phy Control operation. For a Phy Control Write operation, this is the data being written. For a Phy Control Read operation where Phy Control Pending, Phy Control Error and RRAP Abort are all Clear, this is the contents of the register described by Phy Location, UpperAddr and LowerAddr. For a Phy Control Read operation where any of Phy Control Pending, Phy Control Error or RRAP Abort are Set, this field is undefined. Behavior is undefined if this register is written and Phy Control Pending is Set.	RW
28:8	Reserved	RsvdP

Bit Location	Register Description	Attributes
29	**Phy Control Error** – This bit is Cleared when a Phy Control operation is started and is Set when a Phy Control operation completes in error. Possible errors are: The LTSSM is not in the Configuration.Software state (i.e. M-PCIe Configuration.Software Status is 0b). Phy Location is 0 and Upper Addr / Lower Addr refer to a register that does not exist. Phy Location is 1 and an RRAP Request took longer than tRRAPInitiatorResponse (see Table 8-5). Response packet was received with 8b/10b error or parity error (see Section 8.2.3) Received unexpected response (see Section 8.2.3) Default 0b.	RO
30	**RRAP Abort (A)** – If the most recent Phy Control operation was a remote request (Phy Location 1), this bit contains the value of the A bit in the associated RRAP Response packet. If a Phy Control operation is pending or the most recent Phy Control operation was local (Phy Location 0), this bit is Clear. Default 0b.	RO
31	**Phy Control Pending** – This bit is Set when a Phy Control operation is started and is Cleared when that operation completes. Default 0b.	RO

7.29. PASID Extended Capability Structure

Comment [VS311]: ECN:PASID

The presence of a PASID Extended Capability indicates that the Endpoint supports sending and receiving TLPs containing a PASID TLP Prefix. Separate support and enables are provided for the various optional features.

This capability is applicable to Endpoints and Root Complex Integrated Endpoints. For Root Ports, support and control is outside the scope of this specification.

This capability is independent of both the ATS and PRI features defined in the *Address Translation Services Specification*. Endpoints that contain a PASID Extended Capability need not support ATS or PRI. Endpoints that support ATS or PRI need not support PASID.

Figure 7-138 details allocation of the register bits in the PASID Extended Capability structure.

Field Code Changed

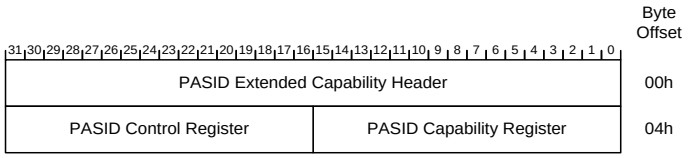


Figure 7-138: PASID Extended Capability Structure

7.29.1. PASID Extended Capability Header (Offset 00h)

Figure 7-139 details allocation of the register fields in the PASID Extended Capability header; Table 7-117 provides the respective bit definitions.

Field Code Changed

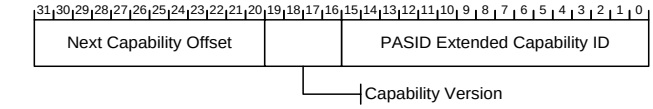


Figure 7-139: PASID Extended Capability Header

Table 7-125: PASID Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PASID Extended Capability ID** – Indicates the PASID Extended Capability structure. This field must return a Capability ID of 001Bh indicating that this is a PASID Extended Capability structure.	RO

Bit Location	Register Description	Attributes
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – The offset to the next PCI Extended Capability structure or 000h if no other items exist in the linked list of capabilities.	RO

7.29.2. PASID Capability Register (Offset 04h)

Figure 7-140 details the allocation of register bits of the PASID Capability register; Table 7-118 provides the respective bit definitions.

Field Code Changed

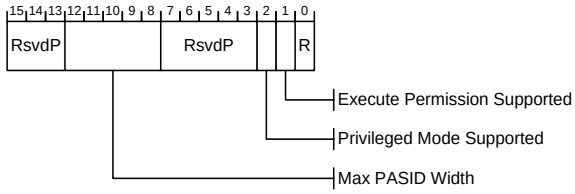


Figure 7-140: PASID Capability Register

Table 7-126: PASID Capability Register

Bit Location	Register Description	Attributes
0	**Reserved**	RsvdP
1	**Execute Permission Supported** – If Set, the Endpoint supports sending TLPs that have the Execute Requested bit Set. If Clear, the Endpoint will never Set the Execute Requested bit.	RO
2	**Privileged Mode Supported** – If Set, the Endpoint supports operating in Privileged and Non-Privileged modes, and supports sending requests that have the Privileged Mode Requested bit Set. If Clear, the Endpoint will never Set the Privileged Mode Requested bit.	RO
7:3	**Reserved**	RsvdP
12:8	**Max PASID Width –** Indicates the width of the PASID field supported by the Endpoint. The value n indicates support for PASID values 0 through 2^n-1 (inclusive). The value 0 indicates support for a single PASID (0). The value 20 indicates support for all PASID values (20 bits). This field must be between 0 and 20 (inclusive).	RO

7.29.3. PASID Control Register (Offset 06h)

Figure 7-141 details the allocation of register bits of the PASID Control register; Table 7-119 provides the respective bit definitions.

Field Code Changed

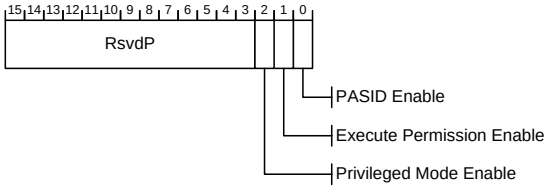


Figure 7-141: PASID Control Register

Table 7-127: PASID Control Register

Bit Location	Register Description	Attributes
0	**PASID Enable –** If Set, the Endpoint is permitted to send and receive TLPs that contain a PASID TLP Prefix. If Clear, the Endpoint is not permitted to do so. Behavior is undefined if the Endpoint supports ATS and this bit changes value when ATS Enable (E) is Set (see the *Address Translation Specification*). Default is 0b.	RW
1	**Execute Permission Enable –** If Set, the Endpoint is permitted to send Requests that have Execute Requested Set. If Clear, the Endpoint is not permitted to do so. Behavior is undefined if the Endpoint supports ATS and this bit changes value when ATS Enable is Set (see the *Address Translation Specification*). If Execute Permission Supported is Clear, this bit is RsvdP. Default is 0b.	RW / RsvdP (see description)
2	**Privileged Mode Enable –** If Set, the Endpoint is permitted to send Requests that have the Privileged Mode Requested Set. If Clear, the Endpoint is not permitted to do so. Behavior is undefined if the Endpoint supports ATS and this bit changes value when ATS Enable is Set (see the *Address Translation Specification*). If Privileged Mode Supported is Clear, this bit is RsvdP. Default is 0b.	RW / RsvdP (see description)

7.30. LNR Extended Capability

Comment [VS312]: ECN: Lightweight Notification (LN) Protocol

The LN Requester (LNR) Extended Capability is an optional normative capability for Endpoints. All Endpoints that support LN protocol as a Requester must implement this capability. See Section 6.21. This capability may be implemented by any type of Endpoint, but not by any other Function type.

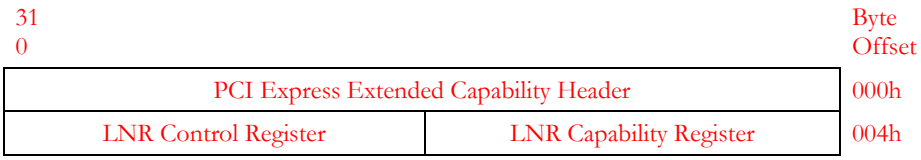
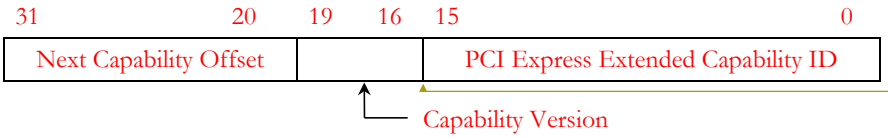


Figure 7-146: LNR Extended Capability

7.30.1. LNR Extended Capability Header (Offset 00h)



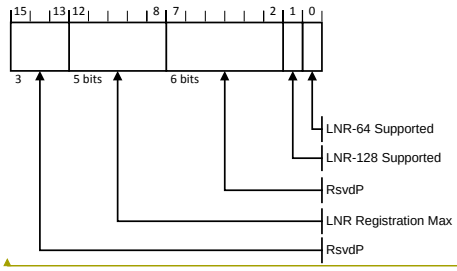
Formatted: Font: 10 pt

Figure 7-147: LNR Extended Capability Header

Table 7-128: LNR Extended Capability Header

Bit Location	Register Description	Attributes
15:0	PCI Express Extended Capability ID – This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. PCI Express Extended Capability ID for the LNR Extended Capability is 001Ch.	RO
19:16	Capability Version – This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 1h for this version of the specification.	RO
31:20	Next Capability Offset – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of capabilities.	RO

7.30.2. LNR Capability Register (Offset 04h)



Formatted: Font: 8 pt

Figure 7-148: LNR Capability Register

Table 7-129: LNR Capability Register

Bit Location	Register Description	Attributes
0	LNR-64 Supported – This bit must be 1b if the Endpoint supports LN protocol for 64-byte cachelines as a Requester; otherwise, must be 0b. See Section 6.21.4 for additional details.	RO
1	LNR-128 Supported – This bit must be 1b if the Endpoint supports LN protocol for 128-byte cachelines as a Requester; otherwise, must be 0b.	RO
12:8	LNR Registration Max – This field, encoded as a power of 2, indicates the maximum number of cachelines that this LN Requester is capable of registering concurrently. For example, a value of 00101b indicates that the LN Requester might be capable of registering up to 32 cachelines (2^5) concurrently, and is capable of registering greater than 16.	RO

7.30.3. LNR Control Register (Offset 04h)

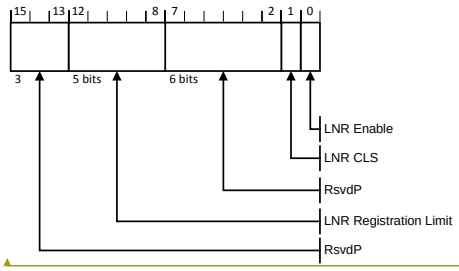


Figure 7-149: LNR Control Register

Table 7-130: LNR Control Register

Bit Location	Register Description	Attributes
0	LNR Enable – When this bit is Set, the Endpoint is enabled to operate as an LN Requester. Software is permitted to Clear this bit at any time. See Section 6.21.4 for requirements regarding the LNR's internal registration state. Default value of this bit is 0b.	RW
1	LNR CLS – This bit controls or indicates the cacheline size used with LN protocol by this Requester. See Section 6.21.4 for restrictions on setting and modifying this bit. If this bit is Clear, the cacheline size is 64 bytes. If this bit is Set, the cacheline size is 128 bytes. If this LN Requester supports only one cacheline size, this bit is permitted to be hardwired to indicate that size. Otherwise, the default value of this bit is 0b.	RW
12:8	LNR Registration Limit – This field, encoded as a power of 2, imposes a limit on the number of cachelines that this LN Requester is permitted to register concurrently. For example, a value of 00100b indicates that the LN Requester must not register more than 16 cachelines (2^4) concurrently. See Section 6.21.4 for restrictions on modifying this field. The default value of this field is 11111b.	RW

7.31. DPC Extended Capability

The Downstream Port Containment (DPC) Extended Capability is an optional normative capability that provides a mechanism for Downstream Ports to contain uncorrectable errors and enable software to recover from them. See Section 6.2.10. This capability may be implemented by a Root Port or a Switch Downstream Port. It is not applicable to any other Device/Port type.

Formatted: Font: 8 pt

Comment [VS313]: ECN: Downstream Port containment (DPC)

If a Downstream Port implements the DPC Extended Capability, that Port must also be capable of reporting the DL_Active state, and indicate so by Setting the Data Link Layer Link Active Reporting Capable bit in the Link Capabilities register. See Section 7.8.6.

The various RP PIO registers must be implemented only by Root Ports that support RP Extensions for DPC, as indicated by the RP Extensions for DPC bit in the DPC Capability register.

Comment [VS314]: ECN: Enhanced DPC

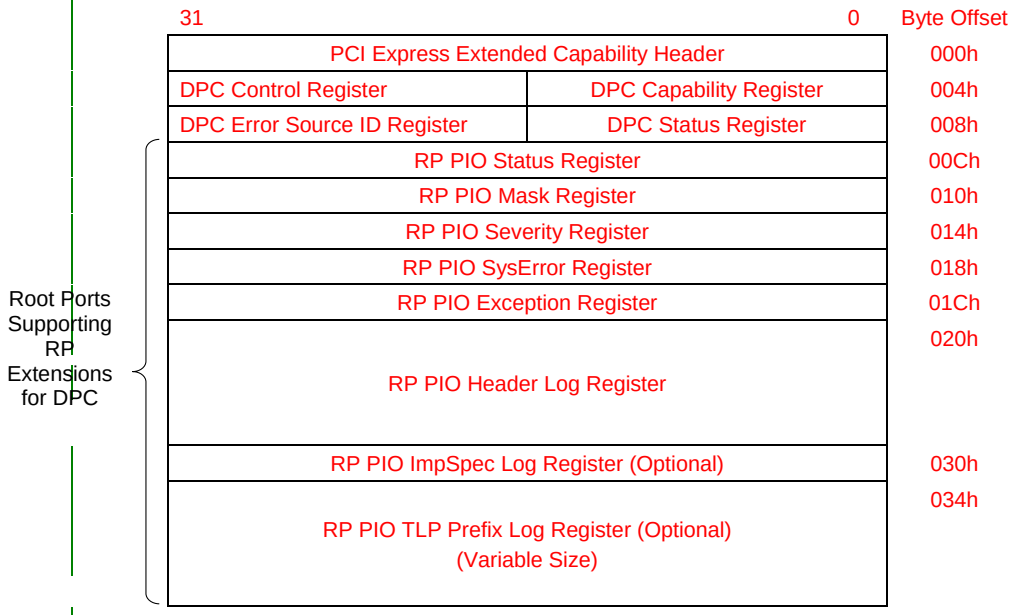


Figure 7-150: DPC Extended Capability

7.31.1. DPC Extended Capability Header (Offset 00h)

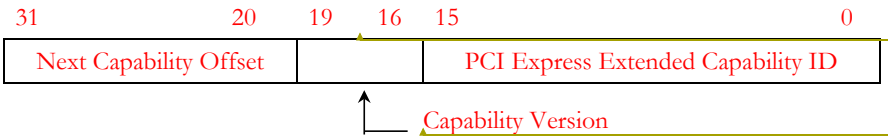


Figure 7-151: DPC Extended Capability Header

Table 7-131: DPC Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. PCI Express Extended Capability ID for the DPC Extended Capability is 001Dh.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of capabilities.	RO

Comment [VS315]: ECN: Downstream Port containment (DPC)

Formatted: Font: Arial, 10 pt

Formatted: Font: 10 pt

Comment [VS316]: ECN: Downstream Port containment (DPC)

7.31.2. DPC Capability Register (Offset 04h)

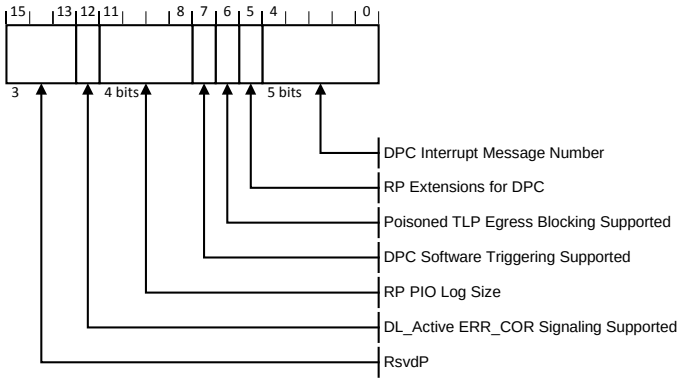


Figure 7-152: DPC Capability Register

Table 7-132: DPC Capability Register

Bit Location	Register Description	Attributes
4:0	**DPC Interrupt Message Number** – This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with the DPC Capability structure. For MSI, the value in this field indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the MSI Message Control register. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined.	RO
5	**RP Extensions for DPC** – If Set, this bit indicates that a Root Port supports a defined set of DPC Extensions that are specific to Root Ports. Switch Downstream Ports must not Set this bit.	RO

Bit Location	Register Description	Attributes
6	**Poisoned TLP Egress Blocking Supported** – If Set, this bit indicates that the Root Port or Switch Downstream Port supports the ability to block the transmission of a poisoned TLP from its Egress Port. Root Ports that support RP Extensions for DPC must Set this bit.	RO
7	**DPC Software Triggering Supported** – If Set, this bit indicates that a Root Port or Switch Downstream Port supports the ability for software to trigger DPC. Root Ports that support RP Extensions for DPC must Set this bit.	RO
11:8	**RP PIO Log Size** – This field indicates how many DWORDs are allocated for the RP PIO log registers, comprised by the RP PIO Header Log, the RP PIO ImpSpec Log, and RP PIO TLP Prefix Log. If the Root Port supports RP Extensions for DPC, the value of this field must be 4 or greater; otherwise, the value of this field must be 0. See Sections 7.31.11, 7.31.12, and 7.31.13.	RO
12	**DL_Active ERR_COR Signaling Supported –** If Set, this bit indicates that the Root Port or Switch Downstream Port supports the ability to signal with ERR_COR when the Link transitions to the DL_Active state. Root Ports that support RP Extensions for DPC must Set this bit.	RO

Comment [VS317]: ECN: Downstream Port containment (DPC)

7.31.3. DPC Control Register (Offset 06h)

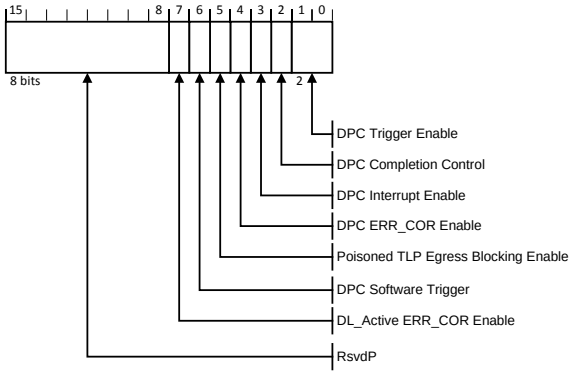


Figure 7-153: DPC Control Register

Table 7-133: DPC Control Register

Bit Location	Register Description	Attributes
1:0	**DPC Trigger Enable** – This field enables DPC and controls the conditions that cause DPC to be triggered. Defined encodings are: 00b DPC is disabled 01b DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_FATAL Message. 10b DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_NONFATAL or ERR_FATAL Message. 11b Reserved Default value of this field is 00b.	RW
2	**DPC Completion Control** – This bit controls the Completion Status for Completions formed during DPC. See Section 2.9.3. Defined encodings are: 0b Completer Abort (CA) Completion Status 1b Unsupported Request (UR) Completion Status Default value of this bit is 0b.	RW
3	**DPC Interrupt Enable** – When Set, this bit enables the generation of an interrupt to indicate that DPC has been triggered. See Section 6.2.10.1. Default value of this bit is 0b.	RW

Bit Location	Register Description	Attributes
4	**DPC ERR_COR Enable** – When Set, this bit enables the sending of an ERR_COR Message to indicate that DPC has been triggered. See Section 6.2.10.2. Default value of this bit is 0b.	RW
5	**Poisoned TLP Egress Blocking Enable** – This bit must be RW if the Poisoned TLP Egress Blocking Supported bit is Set; otherwise, it is permitted to be hardwired to 0b. Software must not Set this bit unless the Poisoned TLP Egress Blocking Supported bit is Set. When Set, this bit enables the associated Egress Port to block the transmission of poisoned TLPs. See Section 2.7.2.2. Default value of this bit is 0b.	RW / RO
6	**DPC Software Trigger** – This bit must be RW if the DPC Software Triggering Supported bit is Set; otherwise, it is permitted to be hardwired to 0b. If DPC is enabled and the DPC Trigger Status bit is Clear, when software writes 1b to this bit, DPC is triggered. Otherwise, software writing a 1b to this bit has no effect. It is permitted to write 1b to this bit while simultaneously writing updated values to other fields in this register, notably the DPC Trigger Enable field. For this case, the DPC Software Trigger semantics are based on the updated value of the DPC Trigger Enable field. This bit always returns 0b when read.	RW / RO
7	**DL_Active ERR_COR Enable** – This bit must be RW if the DL_Active ERR_COR Signaling Supported bit is Set; otherwise, it is permitted to be hardwired to 0b. Software must not Set this bit unless the DL_Active ERR_COR Signaling Supported bit is Set. When Set, this bit enables the associated Downstream Port to signal with ERR_COR when the Link transitions to the DL_Active state. See Section 6.2.10.5. Default value of this bit is 0b.	RW / RO

7.31.4. DPC Status Register (Offset 08h)

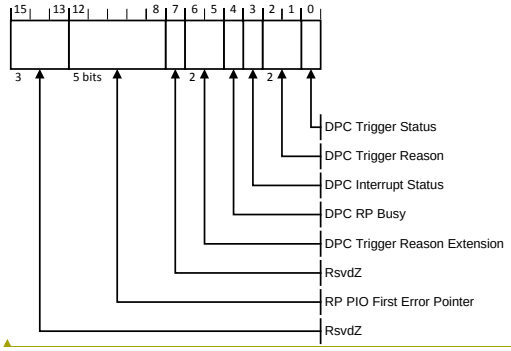


Figure 7-154. DPC Status Register

Table 7-134. DPC Status Register

Bit Location	Register Description	Attributes
0	**DPC Trigger Status** – When Set, this bit indicates that DPC has been triggered. DPC is event triggered. While this bit is Set, hardware must direct the LTSSM to the Disabled State. This bit must be cleared before the LTSSM can be released from the Disabled State. See Section 6.2.10 for requirements on how long software must leave the Downstream Port in DPC. Once these requirements are met, software is permitted to clear this bit regardless of the state of other status bits associated with the triggering event. After clearing this bit, software must honor timing requirements defined in Section 6.6.1 with respect to the first Configuration Read following a Conventional Reset. Default value of this bit is 0b.	RW1CS
2:1	**DPC Trigger Reason** – This field indicates why DPC has been triggered. Defined encodings are: 00b DPC was triggered due to an unmasked uncorrectable error. 01b DPC was triggered due to receiving an ERR_NONFATAL. 10b DPC was triggered due to receiving an ERR_FATAL. 11b DPC was triggered due to a reason that is indicated by the DPC Trigger Reason Extension field. This field is valid only when the DPC Trigger Status bit is Set; otherwise the value of this field is undefined.	ROS
3	**DPC Interrupt Status** – This bit is Set if DPC is triggered while the DPC Interrupt Enable bit is Set. This may cause the generation of an interrupt. See Section 6.2.10.1. Default value of this bit is 0b.	RW1CS

Comment [VS318]: ECN: Downstream Port containment (DPC)

Formatted: Font: 4 pt, Font color: Purple

Comment [VS319]: ECN: Enhanced DPC (eDPC)Nov 2012

Bit Location	Register Description	Attributes
4	**DPC RP Busy** – When the DPC Trigger Status bit is Set and this bit is Set, the Root Port is busy with internal activity that must complete before software is permitted to Clear the DPC Trigger Status bit. If software Clears the DPC Trigger Status bit while this bit is Set, the behavior is undefined. This field is valid only when the DPC Trigger Status bit is Set; otherwise the value of this field is undefined. This bit is applicable only for Root Ports that support RP Extensions for DPC, and is Reserved for Switch Downstream Ports. Default value of this bit is undefined.	RO / RsvdZ
6:5	**DPC Trigger Reason Extension** – This field serves as an extension to the DPC Trigger Reason field. When that field is valid and has a value of 11b, this field indicates why DPC has been triggered. Defined encodings are: 00b DPC was triggered due to an RP PIO error. 01b DPC was triggered due to the DPC Software Trigger bit. 10b Reserved 11b Reserved This field is valid only when the DPC Trigger Status bit is Set and the value of the DPC Trigger Reason field is 11b; otherwise the value of this field is undefined.	ROS
12:8	**RP PIO First Error Pointer** – The value of this field identifies a bit position in the RP PIO Status register, and this field is considered valid when that bit is Set. When this field is valid, and software writes a 1b to the indicated RP PIO Status bit (thus clearing it), this field must revert to its default value. This field is applicable only for Root Ports that support RP Extensions for DPC, and otherwise is Reserved. If this field is not Reserved, its default value is 11111b, indicating a permanently Reserved RP PIO Status bit, thus guaranteeing that this field is not considered valid.	ROS / RsvdZ

7.31.5. DPC Error Source ID Register (Offset 0Ah)

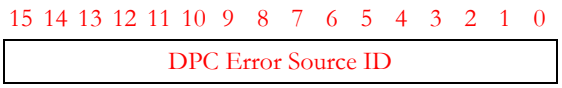


Figure 7-155. DPC Error Source ID Register

Table 7-135. DPC Error Source ID Register

Bit Location	Register Description	Attributes
15:0	**DPC Error Source ID** – When the DPC Trigger Reason field indicates that DPC was triggered due to the reception of an ERR_NONFATAL or ERR_FATAL, this register contains the Requester ID of the received Message. Otherwise, the value of this register is undefined.	ROS

7.31.6. RP PIO Status Register (Offset 0Ch)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

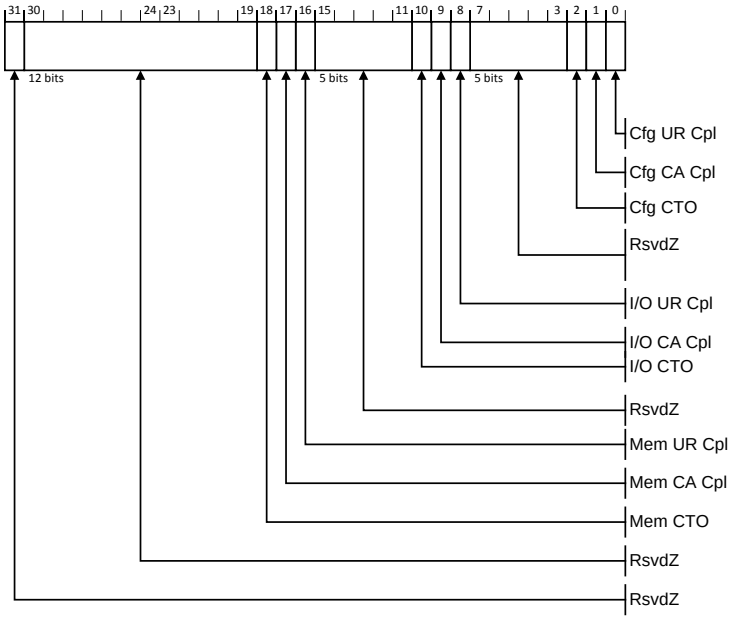


Figure 7-156: RP PIO Status Register

Table 7-136: RP PIO Status Register

Bit Location	Register Description	Attributes	Default
0	Cfg UR Cpl – Configuration Request received UR Completion	RW1CS	0b
1	Cfg CA Cpl – Configuration Request received CA Completion	RW1CS	0b
2	Cfg CTO – Configuration Request Completion Timeout	RW1CS	0b
8	I/O UR Cpl – I/O Request received UR Completion	RW1CS	0b
9	I/O CA Cpl – I/O Request received CA Completion	RW1CS	0b
10	I/O CTO – I/O Request Completion Timeout	RW1CS	0b
16	Mem UR Cpl – Memory Request received UR Completion	RW1CS	0b
17	Mem CA Cpl – Memory Request received CA Completion	RW1CS	0b
18	Mem CTO – Memory Request Completion Timeout	RW1CS	0b
31	Permanently Reserved, since the default RP PIO First Error Pointer field value points to it.	RsvdZ	0b

7.31.7. RP PIO Mask Register (Offset 10h)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

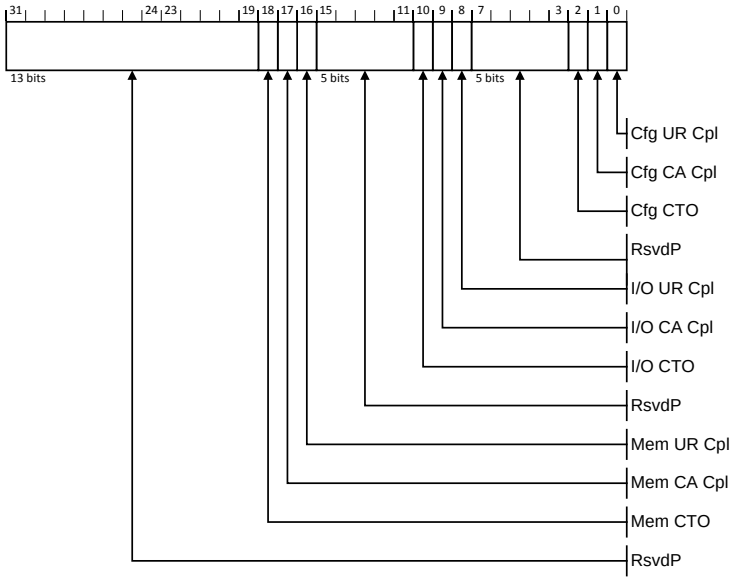


Figure 7-157: RP PIO Mask Register

Table 7-137: RP PIO Mask Register

Bit Location	Register Description	Attributes	Default
0	Cfg UR Cpl – Configuration Request received UR Completion	RWS	1b
1	Cfg CA Cpl – Configuration Request received CA Completion	RWS	1b
2	Cfg CTO – Configuration Request Completion Timeout	RWS	1b
8	I/O UR Cpl – I/O Request received UR Completion	RWS	1b
9	I/O CA Cpl – I/O Request received CA Completion	RWS	1b
10	I/O CTO – I/O Request Completion Timeout	RWS	1b
16	Mem UR Cpl – Memory Request received UR Completion	RWS	1b
17	Mem CA Cpl – Memory Request received CA Completion	RWS	1b
18	Mem CTO – Memory Request Completion Timeout	RWS	1b

7.31.8. RP PIO Severity Register (Offset 14h)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

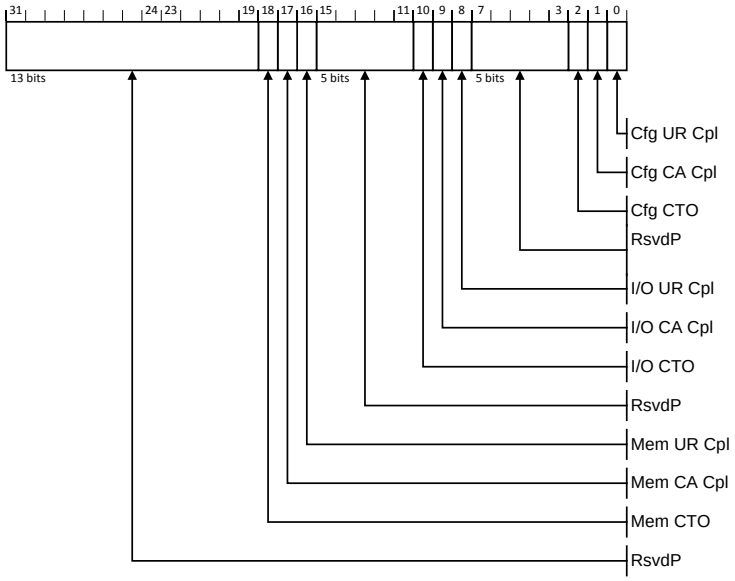


Figure 7-158: RP PIO Severity Register

Table 7-138: RP PIO Severity Register

Bit Location	Register Description	Attributes	Default
0	Cfg UR Cpl – Configuration Request received UR Completion	RWS	0b
1	Cfg CA Cpl – Configuration Request received CA Completion	RWS	0b
2	Cfg CTO – Configuration Request Completion Timeout	RWS	0b
8	I/O UR Cpl – I/O Request received UR Completion	RWS	0b
9	I/O CA Cpl – I/O Request received CA Completion	RWS	0b
10	I/O CTO – I/O Request Completion Timeout	RWS	0b
16	Mem UR Cpl – Memory Request received UR Completion	RWS	0b
17	Mem CA Cpl – Memory Request received CA Completion	RWS	0b
18	Mem CTO – Memory Request Completion Timeout	RWS	0b

7.31.9. RP PIO SysError Register (Offset 18h)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

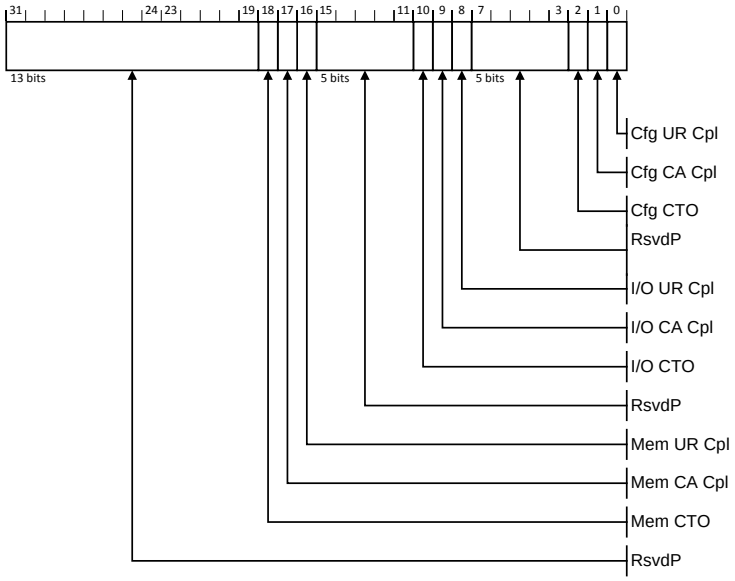


Figure 7-159: RP PIO SysError Register

Table 7-139: RP PIO SysError Register

Bit Location	Register Description	Attributes	Default
0	Cfg UR Cpl – Configuration Request received UR Completion	RWS	0b
1	Cfg CA Cpl – Configuration Request received CA Completion	RWS	0b
2	Cfg CTO – Configuration Request Completion Timeout	RWS	0b
8	I/O UR Cpl – I/O Request received UR Completion	RWS	0b
9	I/O CA Cpl – I/O Request received CA Completion	RWS	0b
10	I/O CTO – I/O Request Completion Timeout	RWS	0b
16	Mem UR Cpl – Memory Request received UR Completion	RWS	0b
17	Mem CA Cpl – Memory Request received CA Completion	RWS	0b
18	Mem CTO – Memory Request Completion Timeout	RWS	0b

7.31.10. RP PIO Exception Register (Offset 1Ch)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

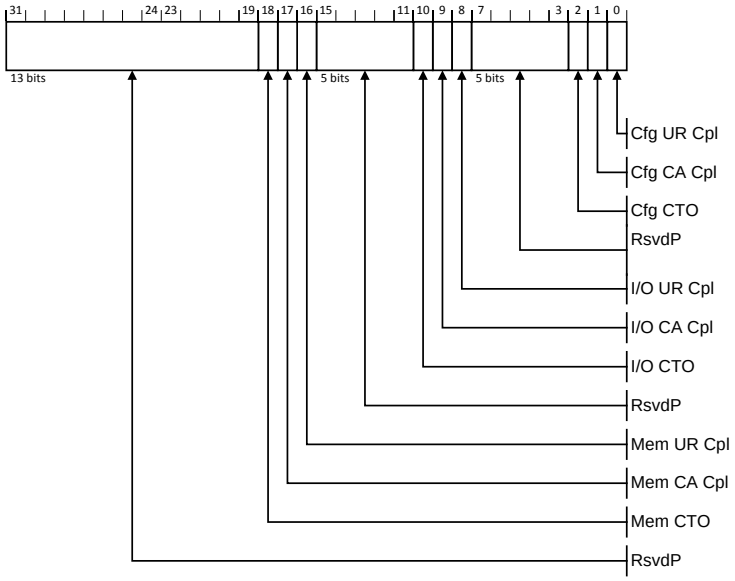


Figure 7-160: RP PIO Exception Register

Table 7-140: RP PIO Exception Register

Bit Location	Register Description	Attributes	Default
0	Cfg UR Cpl – Configuration Request received UR Completion	RWS	0b
1	Cfg CA Cpl – Configuration Request received CA Completion	RWS	0b
2	Cfg CTO – Configuration Request Completion Timeout	RWS	0b
8	I/O UR Cpl – I/O Request received UR Completion	RWS	0b
9	I/O CA Cpl – I/O Request received CA Completion	RWS	0b
10	I/O CTO – I/O Request Completion Timeout	RWS	0b
16	Mem UR Cpl – Memory Request received UR Completion	RWS	0b
17	Mem CA Cpl – Memory Request received CA Completion	RWS	0b
18	Mem CTO – Memory Request Completion Timeout	RWS	0b

7.31.11. RP PIO Header Log Register (Offset 20h)

This register is implemented only in Root Ports that support RP Extensions for DPC. The RP PIO Header Log register contains the header from the TLP associated with a recorded RP PIO error. Refer to Section 6.2.10.3 for further details. This register is 16 bytes and is formatted identically to the Header Log register in AER. See Section 7.10.8.

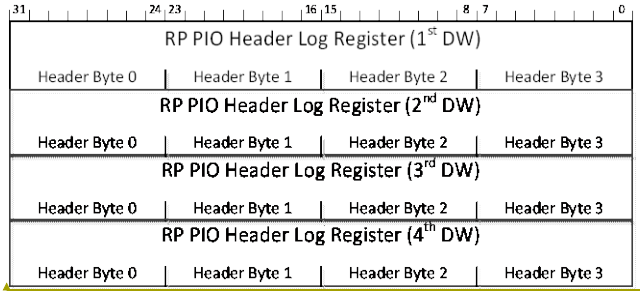


Figure 7-161: RP PIO Header Log Register

Table 7-141: RP PIO Header Log Register

Bit Location	Register Description	Attributes	Default
127:0	Header of TLP associated with error	ROS	0

7.31.12. RP PIO ImpSpec Log Register (Offset 30h)

This register is permitted to be implemented only in Root Ports that support RP Extensions for DPC. The RP PIO ImpSpec Log register, if implemented, contains implementation-specific information associated with the recorded error, e.g., indicating the source of the Request TLP. Space is allocated for this register if the value of the RP PIO Log Size field is 5 or greater. If space is allocated for the register, but the register is not implemented, the bits must be hardwired to 0b.

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
RP PIO ImpSpec Log

Figure 7-162: RP PIO ImpSpec Log Register

Table 7-142: RP PIO ImpSpec Log Register

Bit Location	Register Description	Attributes	Default
31:0	RP PIO ImpSpec Log	ROS	0

Formatted: Font color: Black

7.31.13. RP PIO TLP Prefix Log Register (Offset 34h)

This register is permitted to be implemented only in Root Ports that support RP Extensions for DPC. The RP PIO TLP Prefix Log register contains any End-End TLP Prefixes from the TLP corresponding to a recorded RP PIO error. Refer to Section 6.2.10.3 for further details.

If the Root Port supports tracking Non-Posted Requests that contain End-End TLP Prefixes, this register must be implemented, and must be of sufficient size to record the maximum number of End-End TLP Prefixes for any tracked Request. See Section 2.9.3. The allocated size in DWORDs of the RP PIO TLP Prefix Log register is the RP PIO Log Size minus 5 if the RP PIO Log Size is 9 or less, or 4 if the RP PIO Log Size is greater than 9. The implemented size of the TLP Prefix Log must be less than or equal to the Root Port's Max End-End TLP Prefixes field value. For the case where the Root Port never transmits Non-Posted Requests containing End-End TLP Prefixes, the allocated and implemented size of the TLP Prefix Log is permitted to be 0. Any DWORDs allocated but not implemented must be hardwired to zero.

This register is formatted identically to the TLP Prefix Log register in AER, although this register's allocated size is variable, whereas the register in AER is always 4 DWORDs. See Section 7.10.12. The First TLP Prefix Log register contains the first End-End TLP Prefix from the TLP, the Second TLP Prefix Log register contains the second End-End TLP Prefix, and so forth. If the TLP contains fewer TLP Prefixes than this register accommodates, any remaining TLP Prefix Log registers must contain zero.

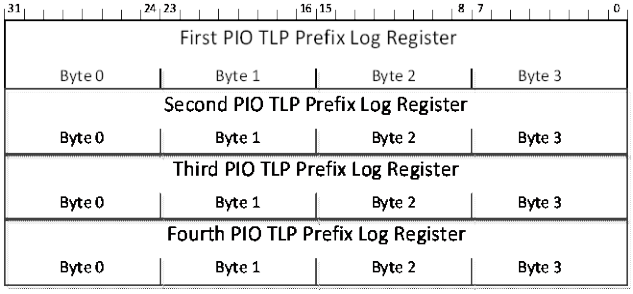


Figure 7-163: RP PIO TLP Prefix Log Register

Table 7-143: RP PIO TLP Prefix Log Register

Bit Location	Register Description	Attributes	Default
127:0	RP PIO TLP Prefix Log	ROS	0

Comment [VS320]: ECN: Precision Time Measurement (PTM), Revision 1.0a

7.32. Precision Time Management (PTM) Capability

The Precision Time Measurement (PTM) Capability is an optional Extended Capability for discovering and controlling the distribution of a PTM Hierarchy. For Root Complexes, this Capability is required in any Root Port, Root Complex Integrated Endpoint, or RCRB that supports PTM. For Endpoints and Switch Upstream Ports that support PTM, this Capability is required in exactly one Function of the Upstream Port and that Capability controls the PTM behavior of all PTM capable Functions associated with that Upstream Port. For Switch Downstream Ports, PTM behavior is controlled by the same PTM Capability that controls the associated Switch Upstream Port. The PTM Capability is not permitted in Bridges, Switch Downstream Ports, and Root Complex Event Collectors.

For Switches, a single instance of this Capability controls behavior for the entire Switch. If the Upstream Port of the Switch is associated with an MFD, it is not required that the controlling Function be the Function corresponding to the Switch Upstream Port. For a given Switch, if this Capability is present, all Downstream Ports of the Switch must implement the requirements defined in Section 6.22.3.2.

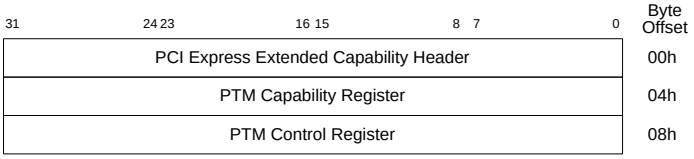


Figure 7-164: PTM Extended Capability Structure

7.32.1. PTM Extended Capability Header (Offset 00h)

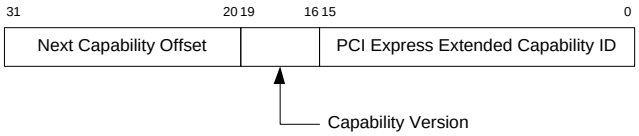


Figure 7-165: PTM Extended Capability Header

Table 7-144: PTM Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCI Express Extended Capability ID for the Precision Time Measurement Capability is 001Fh.	HwInit
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	HwInit
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of Capabilities.	HwInit

7.32.2. PTM Capability Register (Offset 04h)

This register describes a Function's support for Precision Time Measurement. Not all fields within this register apply to all Functions capable of implementing PTM.

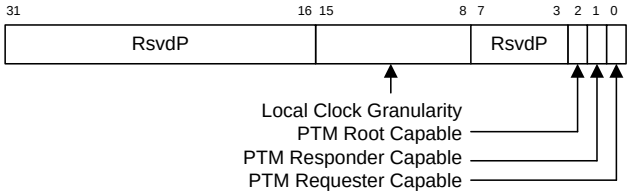


Figure 7-166: PTM Capability Register

Table 7-145: PTM Capability Register

Bit Location	Register Description	Attributes
0	**PTM Requester Capable** – Endpoints and Root Complex Integrated Endpoints are permitted to, and Switches supporting PTM must, set this bit to 1b to indicate they implement the PTM Requester role (see Section 6.22.3.1).	HwInit
1	**PTM Responder Capable** – Root Ports and RCRB's are permitted to, and Switches supporting PTM must, set this bit to 1b to indicate they implement the PTM Responder role (see Section 6.22.3.2). If PTM Root Capable is Set, this bit must be Set to 1b.	HwInit
2	**PTM Root Capable** – Root Ports, RCRB's and Switches are permitted to set this bit to 1b, if they implement a PTM Time Source Role and are capable of serving as the PTM Root.	HwInit
7:3	Reserved	RsvdP
15:8	Local Clock Granularity – 0000 0000b – Time Source does not implement a local clock. It simply propagates timing information obtained from further Upstream in the PTM Hierarchy when responding to PTM Request messages. 0000 0001b – 1111 1110b: Indicates the period of this Time Source's local clock in ns. 1111 1111b: Indicates the period of this Time Source's local clock is greater than 254 ns. If the PTM Root Select bit is Set, this local clock is used to provide PTM Master Time. Otherwise, the Time Source uses this local clock to locally track PTM Master Time received from further Upstream within a PTM Hierarchy. This field is reserved for Functions that do not implement the PTM Time Source role.	HwInit/RsvdP
31:16	Reserved	RsvdP

7.32.3. PTM Control Register (Offset 08h)

This register controls a Function's participation in the Precision Time Measurement mechanism. Not all fields within this register apply to all Functions capable of implementing PTM.

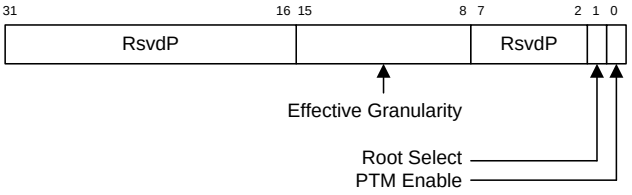


Figure 7-167: PTM Control Register

Table 7-146: PTM Control Register

Bit Location	Register Description	Attributes
0	**PTM Enable** – When Set, this Function is permitted to participate in the PTM mechanism according to its selected role(s) (see Section 6.22.2). Default value is 0b.	RW
1	**Root Select** – When Set, if the PTM Enable bit is also Set, this Time Source is the PTM Root. Within each PTM Hierarchy, it is recommended that system software select only the furthest Upstream Time Source to be the PTM Root. Default value is 0b. If the value of the PTM Root Capable bit is 0b, this bit is permitted to be hardwired to 0b.	RW/RO
7:2	Reserved	RsvdP

Bit Location	Register Description	Attributes
15:8	**Effective Granularity** – For Functions implementing the PTM Requester Role, this field provides information relating to the expected accuracy of the PTM clock, but does not otherwise affect the PTM mechanism. For Endpoints, system software must program this field to the value representing the maximum Local Clock Granularity reported by the PTM Root and all intervening PTM Time Sources. For Root Complex Integrated Endpoints, system software must set this field to the value reported in the Local Clock Granularity field by the associated PTM Time Source. Permitted values: 0000 0000b: Unknown PTM granularity – one or more Switches between this Function and the PTM Root reported a Local Clock Granularity value of 0000 0000b. 0000 0001b – 1111 1110b: Indicates the effective PTM granularity in ns. 1111 1111b: Indicates the effective PTM granularity is greater than 254 ns. Default value is 00000b. If PTM Requester Capable is Clear, this field is permitted to be hardwired to 0000 0000b.	RW/RO
31:16	Reserved	RsvdP

7.33. L1 PM Substates Extended Capability

Comment [VS321]: New section 7.32 added from ECN: L1 PM Substates with CLKREQ

The PCI Express L1 PM Substates Capability is an optional Extended Capability, that is required if L1 PM Substates is implemented at a Port. The L1 PM Substates Extended Capability structure is defined as shown in Figure 7-168.

For a multi-Function device associated with an Upstream Port implementing L1 PM Substates, this Extended Capability Structure must be implemented only in Function 0, and must control the Upstream Port's Link behavior on behalf of all the Functions of the device.

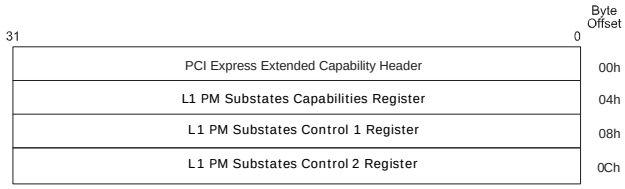


Figure 7-168: L1 PM Substates Capability

7.33.1. L1 PM Substates Extended Capability Header (Offset 00h)

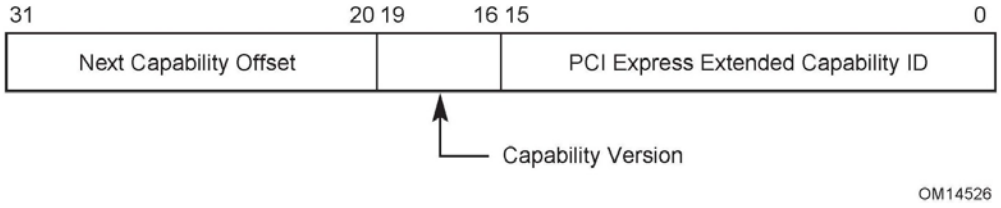


Figure 7-169: L1 PM Substates Extended Capability Header

Table 7-147: L1 PM Substates Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for L1 PM Substates is 001Eh.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO

Bit Location	Register Description	Attributes
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh. The bottom 2 bits of this offset are Reserved and must be implemented as 00b although software must mask them to allow for future uses of these bits.	RO

7.33.2. L1 PM Substates Capabilities Register (Offset 04h)

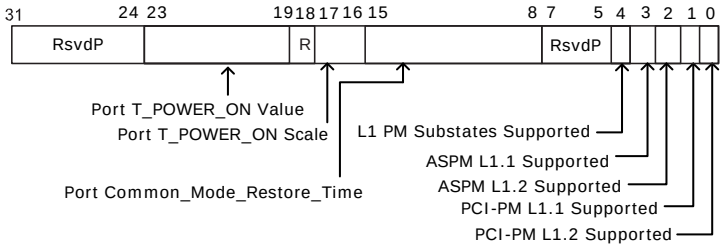


Figure 7-170: L1 PM Substates Capabilities Register

Table 7-148: L1 PM Substates Capabilities Register

Bit Location	Register Description	Attributes
0	**PCI-PM L1.2 Supported** – When Set this bit indicates that PCI-PM L1.2 is supported.	HwInit
1	**PCI-PM L1.1 Supported -** When Set this bit indicates that PCI-PM L1.1 is supported, and must be Set by all Ports implementing L1 PM Substates.	HwInit
2	**ASPM L1.2 Supported** – When Set this bit indicates that ASPM L1.2 is supported.	HwInit
3	**ASPM L1.1 Supported -** When Set this bit indicates that ASPM L1.1 is supported.	HwInit
4	**L1 PM Substates Supported** – When Set this bit indicates that this Port supports L1 PM Substates.	HwInit
7:5	**Reserved**	RsvdP

Bit Location	Register Description	Attributes
15:8	**Port Common_Mode_Restore_Time –** Time (in µs) required for this Port to re-establish common mode as described in Table 5-11. Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP.	HwInit/RsvdP (See description)
17:16	**Port T_POWER_ON Scale –** Specifies the scale used for the Port T_POWER_ON Value field in the L1 PM Substates Capabilities register. Range of Values 00b = 2 µs 01b = 10 µs 10b = 100 µs 11b = Reserved Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. Default value is 00b	HwInit / RsvdP
18	**Reserved**	RsvdP
23:19	**Port T_POWER_ON Value –** Along with the Port T_POWER_ON Scale field in the L1 PM Substates Capabilities register sets the time (in us) that this Port requires the port on the opposite side of Link to wait in L1.2.Exit after sampling CLKREQ# asserted before actively driving the interface. The value of Port T_POWER_ON is calculated by multiplying the value in this field by the scale value in the Port T_POWER_ON Scale field in the L1 PM Substates Capabilities register. Default value is 00101b Required for all Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP.	HwInit / RsvdP
31:24	**Reserved**	RsvdP

7.33.3. L1 PM Substates Control 1 Register (Offset 08h)

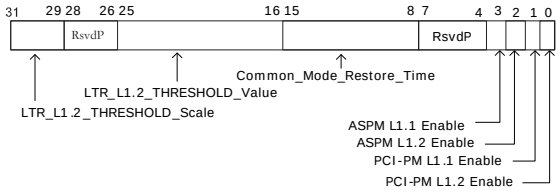


Figure 7-171: L1 PM Substates Control 1 Register

Table 7-149: L1 PM Substates Control 1 Register

Bit Location	Register Description	Attributes
0	**PCI-PM L1.2 Enable** – When Set this bit enables PCI-PM L1.2. Required for both Upstream and Downstream Ports. For Ports for which the PCI-PM L1.2 Supported bit is Clear this bit is permitted to be hardwired to 0. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported bit in the L1 PM Substates Capabilities Register is Set. Default value is 0b.	RW
1	**PCI-PM L1.1 Enable -** When Set this bit enables PCI-PM L1.1. Required for both Upstream and Downstream Ports. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported bit in the L1 PM Substates Capabilities Register is Set. Default value is 0b.	RW
2	**ASPM L1.2 Enable** – When Set this bit enables ASPM L1.2. Required for both Upstream and Downstream Ports. For Ports for which the ASPM L1.2 Supported bit is Clear this bit is permitted to be hardwired to 0. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported bit in the L1 PM Substates Capabilities Register is Set. Default value is 0b.	RW

3	**ASPM L1.1 Enable -** When Set this bit enables ASPM L1.1. Required for both Upstream and Downstream Ports. For Ports for which the ASPM L1.1 Supported bit is Clear this bit is permitted to be hardwired to 0. For compatibility with possible future extensions, software must not enable L1 PM Substates unless the L1 PM Substates Supported bit in the L1 PM Substates Capabilities Register is Set. Default value is 0b.	RW
7:4	**Reserved**	RsvdP
15:8	**Common_Mode_Restore_Time –** Sets value of $T_{COMMONMODE}$ (in µs), which must be used by the Downstream Port for timing the re-establishment of common mode, as described in Table 5-11. This field must only be modified when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Clear. The Port behavior is undefined if this field is modified when either the ASPM L1.2 Enable and/or PCI-PM L1.2 Enable bit(s) are Set. Required for Downstream Ports for which either the PCI-PM L1.2 Supported bit is Set, ASPM L1.2 Supported bit is Set, or both are Set, otherwise this field is of type RsvdP. This field is of type RsvdP for Upstream Ports. Default value is implementation specific.	RW/RsvdP (See Description)
25:16	**LTR_L1.2_THRESHOLD_Value** – Along with the LTR_L1.2_THRESHOLD_Scale, this field indicates the LTR threshold used to determine if entry into L1 results in L1.1 (if enabled) or L1.2 (if enabled). The default value for this field is 00 0000 0000b. This field must only be modified when the ASPM L1.2 Enable bit is Clear. The Port behavior is undefined if this field is modified when the ASPM L1.2 Enable bit is Set. Required for all Ports for which the ASPM L1.2 Supported bit is Set, otherwise this field is of type RsvdP.	RW/RsvdP (See Description)
28:26	**Reserved**	RsvdP

31:29	**LTR_L1.2_THRESHOLD_Scale** – This field provides a scale for the value contained within the LTR_L1.2_THRESHOLD_Value. Encoding is the same as the LatencyScale fields in the LTR Message (see Section 6.18). The default value for this field is 000b. Hardware operation is undefined if software writes a Not-Permitted value to this field. This field must only be modified when the ASPM L1.2 Enable bit is Clear. The Port behavior is undefined if this field is modified when the ASPM L1.2 Enable bit is Set. Required for all Ports Ports for which the ASPM L1.2 Supported bit is Set, otherwise this field is of type RsvdP.	RW/RsvdP (See description)

7.33.4. L1 PM Substates Control 2 Register (Offset 0Ch)

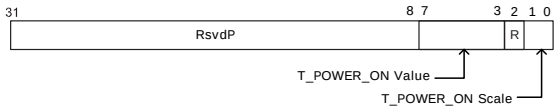


Figure 7-172: L1 PM Substates Control 2 Register

Table 7-150: L1 PM Substates Control 2 Register

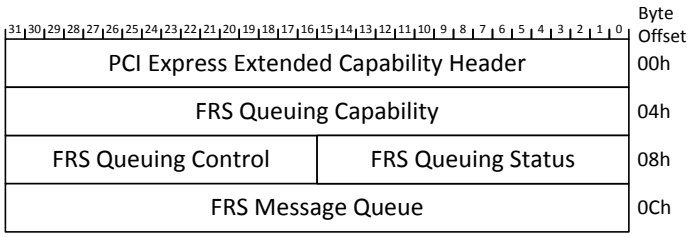
Bit Location	Register Description	Attributes
1:0	**T_POWER_ON Scale –** Specifies the scale used for T_POWER_ON Value. Range of Values 00b = 2 µs 01b = 10 µs 10b = 100 µs 11b = Reserved Required for all Ports that support L1.2, otherwise this field is of type RsvdP. This field must only be modified when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Clear. The Port behavior is undefined if this field is modified when either the ASPM L1.2 Enable and/or PCI-PM L1.2 Enable bit(s) are Set. Default value is 00b	RW/ RsvdP

Bit Location	Register Description	Attributes
2	**Reserved**	RsvdP
7:3	**T_POWER_ON Value –** Along with the T_POWER_ON Scale sets the minimum amount of time (in µs) that the Port must wait in L1.2.Exit after sampling CLKREQ# asserted before actively driving the interface. T_POWER_ON is calculated by multiplying the value in this field by the value in the T_POWER_ON Scale field. This field must only be modified when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Clear. The Port behavior is undefined if this field is modified when either the ASPM L1.2 Enable and/or PCI-PM L1.2 Enable bit(s) are Set. Default value is 00101b Required for all Ports that support L1.2, otherwise this field is of type RsvdP.	RW/ RsvdP
31:8	**Reserved**	RsvdP

7.34. Function Readiness Status (FRS) Queuing Extended Capability

The Function Readiness Status (FRS) Queuing Extended Capability is required for Root Ports and Root Complex Event Collectors that support the optional normative FRS Queuing capability. See Section 6.23. This extended capability is only permitted in Root Ports and Root Complex Event Collectors.

If this capability is present in a Function, that Function must also implement either MSI, MSI-X, or both.



Field Code Changed

Figure 7-173: FRS Extended Capability

7.34.1. Function Readiness Status (FRS) Queuing Extended Capability Header (Offset 00h)

Field Code Changed

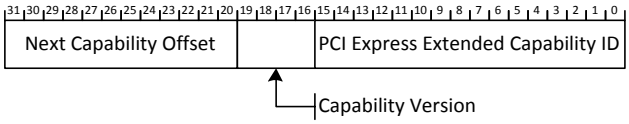


Figure 7-174: FRS Queuing Extended Capability Header

Table 7-151: FRS Queuing Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. PCI Express Extended Capability ID for the FRS Queuing Extended Capability is 0021h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of capabilities.	RO

7.34.2. FRS Queuing Capability Register (Offset 04h)

Field Code Changed

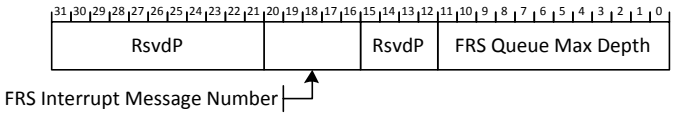


Figure 7-175: FRS Queuing Capability Register

Table 7-152: FRS Queuing Capability Register

Bit Location	Register Description	Attributes
11:0	**FRS Queue Max Depth** – Indicates the implemented queue depth, with valid values ranging from 001h (queue depth of 1) to FFFh (queue depth of 4095) The value of FRS Message Queue Depth must not exceed this value. The value 000h is Reserved.	HwInit
20:16	**FRS Interrupt Message Number** – This register indicates which MSI/MSI-X vector is used for the interrupt message generated in association with FRS Message Received or FRS Message Overflow. For MSI, the value in this register indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the MSI Message Control register. For MSI-X, the value in this register indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this register must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this register must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this register is undefined.	RO

7.34.3. FRS Queuing Status Register (Offset 08h)

Field Code Changed

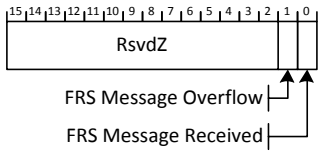


Figure 7-176: FRS Queueing Status Register

Table 7-153: FRS Queuing Status Register

Bit Location	Register Description	Attributes
0	**FRS Message Received** – This bit is Set when a new FRS Message is Received or generated by this Root Port or Root Complex Event Collector. Root Ports must Clear this bit when the Link is DL_Down. Default value of this bit is 0b.	RW1C
1	FRS Message Overflow – This bit is set if the FRS Message queue is full and a new FRS Message is received or generated by this Root Port or Root Complex Event Collector. Root Ports must Clear this bit when the Link is DL_Down. Default value of this bit is 0b.	RW1C
15:2	Reserved	RsvdZ

7.34.4. FRS Queuing Control Register (Offset 0Ah)

Field Code Changed

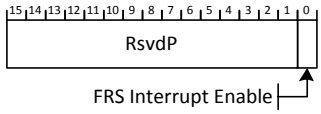


Figure 7-177: FRS Queueing Control Register

Table 7-154: FRS Queuing Control Register

Bit Location	Register Description	Attributes
0	**FRS Interrupt Enable** – When Set and MSI or MSI-X is enabled, the Port must issue an MSI/MSI-X interrupt to indicate the 0b to 1b transition of either the FRS Message Received or the FRS Message Overflow bits. Default value of this bit is 0b.	RW
15:1	Reserved	RsvdP

7.34.5. FRS Message Queue Register (Offset 0Ch)

The FRS Message Queue Register contains fields from the oldest FRS message in the queue. It also indicates the number of FRS messages in the queue.

A write of any value that includes byte 0 to this register removes the oldest FRS Message from the queue and updates these fields. A write to this register when the queue is empty has no effect.

Field Code Changed

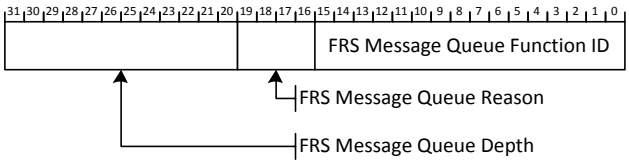


Figure 7-178: FRS Message Queue Register

Table 7-155: FRS Message Queue Register

Bit Location	Register Description	Attributes
15:0	**FRS Message Queue Function ID** – Recorded from the Requester ID of the oldest FRS Message received or generated by this Root Port or Root Complex Event Collector and still in the queue. Undefined if FRS Message Queue Depth is 000h.	RO
19:16	**FRS Message Queue Reason** – Recorded from the FRS Reason of the oldest FRS Message received or generated by this Root Port or Root Complex Event Collector and still in the queue. Undefined if FRS Message Queue Depth is 000h.	RO
31:20	**FRS Message Queue Depth** – indicates the current number of FRS Messages in the queue. The value of 000h indicates an empty queue. Default value of this field is 000h.	RO

7.35. Readiness Time Reporting Extended Capability

The Readiness Time Reporting Extended Capability provides an optional mechanism for describing the time required for a Device or Function to become Configuration-ready. In the indicated situations, software is permitted to issue Requests to the Device or Function after waiting for the time advertised in this capability and need not wait for the (longer) times required elsewhere.

Software is permitted to issue requests upon the earliest of:

- Receiving a Readiness Notifications message (see Section 6.23).
- Waiting the appropriate time as specified in this document or in applicable specifications including the PCI Local Bus Specification, PCI Bus Power Management Interface Specification.
- Waiting the time indicated in the associated field of this capability.
- Waiting the time defined by system software or firmware[125].

Software is permitted to cache values from this capability and to use those cached values when the device topology has not changed.

This capability is permitted to be implemented in all Functions.

A Function must be Configuration-Ready if:

- The Immediate Readiness bit is Clear and at least **Reset Time** has elapsed after the completion of Conventional Reset
 - If the Immediate Readiness bit is Set, **Reset Time** does not apply, and is Reserved
- The Function is associated with an Upstream Port and at least **DL_Up Time** has elapsed after the Downstream Port above that Function reported Data Link Layer Link Active (see Section 7.8.8).
- The Function supports Function Level Reset and at least **FLR Time** has elapsed after that Function was issued a Function Level Reset.
- Immediate_Readiness_on_Return_to_D0 is Clear and at least **$D3_{hot}$ to D0 Time** has elapsed after that Function was directed to the D0 state from $D3_{hot}$.
 - If the Immediate_Readiness_on_Return_to_D0 bit is Set, **$D3_{hot}$ to D0 Time** does not apply, and is Reserved

When Immediate_Readiness_on_Return_to_D0 is Clear, a Function must be Configuration-Ready when at least **$D3_{hot}$ to D0 Time** has elapsed after the Function was directed to the D0 state from $D3_{hot}$. In addition, the Function must be in either the $D0_{uninitialized}$ or $D0_{active}$ state, depending on the value of the No_Soft_Reset bit.

For VFs additional behavior is defined in the *Single Root I/O Virtualization and Sharing Specification*.

[125] For example, using ACPI tables to provide the equivalent of this capability.

If the above conditions do not apply, Function behavior is not determined by the Readiness Time Reporting Extended Capability, and the Function must respond as defined elsewhere (including, for example, no response or a response with Configuration Retry Status).

The time values reported are determined by implementation-specific mechanisms. A valid bit is defined in this capability to permit a device to defer reporting time values, for example to allow hardware initialization through driver-based mechanisms. If the Valid bit remains Clear and 1 minute has elapsed after device driver(s) have started, software is permitted to assume that no values will be reported.

Registers and fields in the Readiness Time Reporting Extended Capability are shown in Figure 7-179. Time values are encoded in floating point as shown in Figure 7-180. The actual time value is $Value \times Multiplier[Scale]$. For example, the value A1Eh represents about 1 second (actually 1.006 sec) and the value 40Ah represents about 10 ms (actually 10.240 ms).

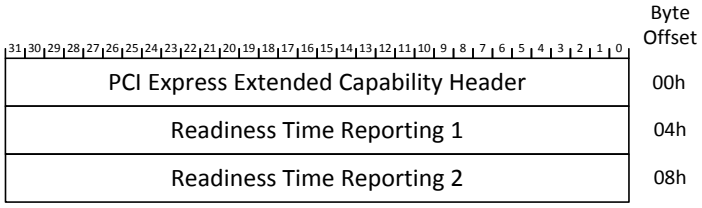
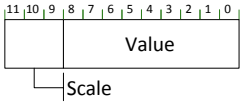


Figure 7-179: Readiness Time Reporting Extended Capability



Scale	Multiplier
0	1 ns
1	32 ns
2	1,024 ns
3	32,768 ns
4	1,048,576 ns
5	33,554,432 ns
6	Reserved
7	Reserved

Multiplier = 32^{scale} ns

Figure 7-180: Readiness Time Encoding

7.35.1. Readiness Time Reporting Extended Capability Header (Offset 00h)

Figure 7-181 and Table 7-156: Readiness Time Reporting Extended Capability Header detail allocation of fields in the Extended Capability header.

Field Code Changed

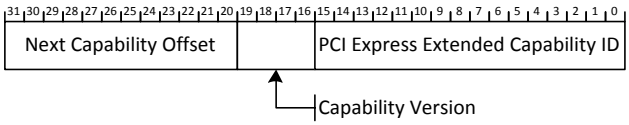


Figure 7-181: Readiness Time Reporting Extended Capability Header

Table 7-156: Readiness Time Reporting Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. Extended Capability ID for the Readiness Time Reporting Extended Capability is 0022h.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of Capabilities. For Extended Capabilities implemented in Configuration Space, this offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating list of Capabilities) or greater than 0FFh.	RO

7.35.2. Readiness Time Reporting 1 (Offset 04h)

Figure 7-182 and Table 7-157: Readiness Time Reporting 1 Register detail allocation of fields in the Readiness Time Reporting 1 Register.

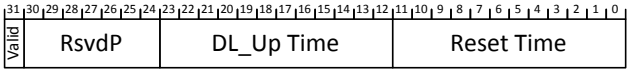


Figure 7-182: Readiness Time Reporting 1 Register

Table 7-157: Readiness Time Reporting 1 Register

Comment [VS322]: Updated per RC_ECN_RN_27_Aug_2013

Bit Location	Register Description	Attributes
11:0	**Reset Time** – is the time the Function requires to become Configuration-Ready after the completion of Conventional Reset. This field is RsvdP if the Immediate Readiness bit is Set. This field is undefined when the Valid bit is Clear. This field must be less than or equal to the encoded value A1Eh.	HwInit/RsvdP
23:12	**DL_Up Time** – is the time the Function requires to become Configuration-Ready after the Downstream Port above the Function reports Data Link Layer Link Active. This field is RsvdP in Functions that are not associated with an Upstream Port. This field is undefined when the Valid bit is Clear. This field must be less than or equal to the encoded value A1Eh.	HwInit/RsvdP
30:24	**Reserved**	RsvdP
31	**Valid** – If Set, indicates that all time values in this capability are valid. If Clear, indicates that the time values in this capability are not yet available. Time values may depend on device configuration. Device specific mechanisms, possibly involving the device driver(s), could be involved in determining time values. If this bit remains Clear and 1 minute has elapsed after all associated device driver(s) have started, software is permitted to assume that this bit will never be set.	HwInit

7.35.3. Readiness Time Reporting 2 (Offset 08h)

Figure 7-183 and Table 7-158: Readiness Time Reporting 2 Register detail allocation of fields in the Readiness Time Reporting 2 Register.

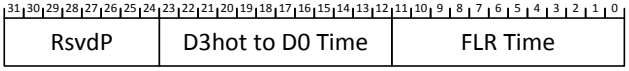


Figure 7-183: Readiness Time Reporting 2 Register

Table 7-158: Readiness Time Reporting 2 Register

Bit Location	Register Description	Attributes
11:0	**FLR Time** – is the time that the Function requires to become Configuration-Ready after it was issued an FLR. This field is RsvdP when the Function Level Reset Capability bit is Clear (see Section 7.8.3). This field is undefined when the Valid bit is Clear. This field must be less than or equal to the encoded value A1Eh.	HwInit/RsvdP
23:12	**$D3_{hot}$ to D_0 Time** –If Immediate_Readiness_on_Return_to_D0 is Clear, $D3_{hot}$ to D0 Time is the time that the Function requires after it is directed from $D3_{hot}$ to D0 before it is Configuration-Ready and has returned to either $D0_{unitialized}$ or $D0_{active}$ state (see the *PCI Bus Power Management Interface Specification*). This field is RsvdP if the Immediate_Readiness_on_Return_to_D0 bit is Set. This field is undefined when the Valid bit is Clear. This field must be less than or equal to the encoded value 40Ah.	HwInit/RsvdP
31:24	**Reserved**	RsvdP

8

8. M-PCIe Logical Sub-Block

Comment [VS323]: Entire Chapter 8 added per ECN: M-PCIe

The M-PCIe logical Sub-Block is shown in Figure 8-1.

Formatted: Font color: Auto

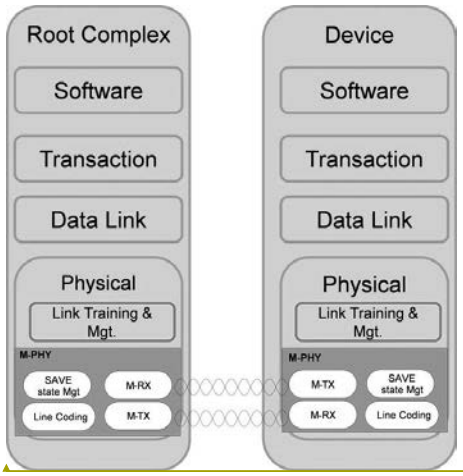


Figure 8-1: Adaptation of MIPI-M-PHY

The terms LINE, LANE, SUB-LINK and LINK, and all M-PHY interface mechanisms are referred to in this chapter using the terminology defined in [M-PHY]. [M-PHY] defined interfaces and related terms must be capitalized.

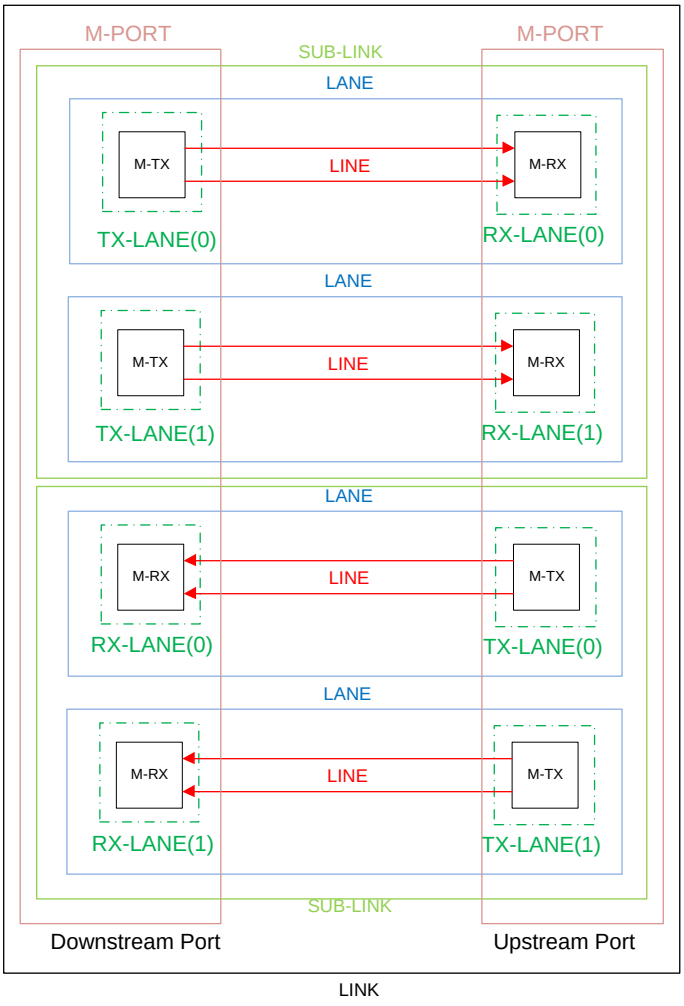


Figure 8-2: M-PCIe LINK Example.

A LINK consists of a Downstream Port and an Upstream Port and the associated M-PORT, SUB-LINKs, LANEs, MODULEs and LINEs (see Figure 8-2).

The TX-LANE(n) on the M-PORT refers to the M-TX MODULE and transmitter differential pair signals TXDP and TXDN for the transmit SUB-LINK, where n corresponds to the LANE number of the M-TX MODULE supported by the transmit SUB-LINK. The RX-LANE(m) on M-PORT refers to the M-RX MODULE and receiver differential pair signals RXDP and RXDN for the receive SUB-LINK, where m corresponds to the LANE number of the M-RX MODULE supported by the receive SUB-LINK.

The minimum LINK configuration must consist of one LANE for the transmit SUB-LINK and one LANE for the receive SUB-LINK. This combination is identified as LANE PAIR0 and is required

for the LINK Initial Discovery and Configuration process. LANE PAIR0 consists of TX-LANE(0) and RX-LANE(0) on Downstream Port and the corresponding RX0-LANE(0) and TX-LANE(0) on Upstream Port.

For any LANE to be "Configured", the TX-LANE(x) must be connected to the corresponding remote RX-LANE(x). The numbering of TX-LANE and RX-LANE must be determined statically by the system integrator through implementation-specific mechanisms.

8.1. PHY Requirements

The following summarizes the required PHY mechanisms and specific M-PHY capabilities:

- Type –I M-PHY
- HS-MODE operation
- Rate A, B series
- LS-MODE for M-PHY attribute initialization and setting TEST mode operation
- Capability discovery and configuration of M-PHY parameters
- Multi-Lane Power Management compliant to PCI Express PCI-PM and ASPM
- Compliant to M-PHY 8b/10b encoding scheme used for spectral conditioning and clock recovery
- Shared or independent reference clocks

The following M-PHY features are not required:

- Type-II MODULEs.
- LS-MODE for run-time operation
- LINE-CFG
- Line Control Command (LCC)
- Optical Media Converters (OMC)

The following rules apply to a SUB-LINK:

- A SUB-LINK must have at least one LANE per SUB-LINK.
- The set of permitted numbers of LANEs per SUB-LINK is: x1, x2, x4, x8, x12, x16 and x32.
- The term Configured LANEs is defined as the number of LANEs that are configured to form a SUB-LINK.
- All the LANEs within a SUB-LINK must be configured to operate at the same HS-GEAR and RATE series.

The following rules apply to a LINK:

- A LINK must have 2 SUB-LINKs.
- SUB-LINKs must be configured to operate at the same HS-GEAR and RATE series.
- SUB-LINKs are permitted to support different widths.

Table 8-1 defines the required speed and rate capabilities:

Table 8-1: Table Speed and Rate Capabilities

M-PCIe Speeds Supported	Required HS-GEAR Support			
	PWM-G1[1]	HS-G1	HS-G2	HS-G3
HS- G1	Required	Required	n/a	n/a
HS-G2	Required	Optional	Required	n/a
HS-G3	Required	Optional	Optional	Required

[1]PWM-G1 is required for initialization and test mechanisms.

8.2. Configuration

The M-PHY Specification defines the capabilities and attributes associated with each M-TX and M-RX MODULE. This information is also included in Section 8.8 of this specification. Section 8.2.2 of this specification defines the capabilities and attributes associated with a M-PCIe Link.

The LINK Discovery and Configuration process is initiated by the Upstream component in Configuration.Start. During this process, the Upstream component determines the capabilities of the Downstream component, the M-PHY MODULE(s) associated with the Downstream Port, and the M-PHY MODULE(s) associated with the Upstream Port. Based on these capabilities, the Downstream Port then configures the appropriate attributes.

The Upstream component uses the Remote Register Access Protocol (RRAP) on LANE 0 to read and write the capabilities and attributes associated with the Upstream Port (including the values of the Port and of the associated M-PHY MODULE(s)).

8.2.1. Link Discovery and Configuration

In the Configuration.Start Link state LANE PAIR0 must be used for inline configuration, where MODULE capabilities can be discovered and configuration settings are applied. The LINK enters LS-MODE after power-on-reset or from Link Down, all LANEs first enter HIBERN8 and only LANE PAIR0 transitions to SLEEP state. The LINK initialization protocol uses LANE PAIR0 for initial configuration and must operate at PWM-G1 data rate only.

During the LS-MODE of operation the Downstream port acts as the master and the Upstream port acts as the target. The requests to read and write parameters are generated by the Downstream port and responses are returned by the Upstream port.

The Initial LINK Discovery and Configuration process consists of the following steps:

- The LINK operates with LANE PAIR0, in LS-MODE, and the following apply:
 - The RRAP packets are not subjected to flow-control, and Data Link level retry mechanisms.
 - Only PAD and COM symbols, and RRAP packets, are permitted to be transmitted in Configuration.Start.
 - RRAP packets must be 8b/10b encoded but not scrambled.

Comment [VS324]: Corrected Bullet levels

- In the Discovery phase the Upstream component reads the Attributes on both sides of the LINK and determines the Attribute configuration settings for both MODULES.
 - Local Register Configuration (LRC) is the process used by the Port to configure the parameters of the Local MODULE. The mechanism by which LRC is implemented is outside the scope of the specification.
 - Remote Register Configuration (RRC) is the process used by the Upstream component to access the capability and configuration attributes of the Downstream component's M-PHY MODULES. The RRAP protocol must be used by the Upstream component to configure the Remote MODULE.
 - Remote LANE attributes must be read and configured using the RRAP protocol.
 - When LINK_CFG_RETAIN_STATUS is Set, this indicates that M-PHY attribute configuration is retained. Upstream component is permitted to use this information to determine if any further M-PHY attribute discovery and configuration is required.
 - At least 1 RRAP command and the corresponding response must be exchanged successfully over the link during the discovery phase.
- In the PHY CONFIG phase the Upstream component, if required, updates the Attribute configuration settings on both sides of the LINK.
- In the EFFECTUATE phase the Configured LANES are transitioned to SAVE state. Upon entry into SAVE State the MODULES must update the Configuration registry.
 - LANE PAIR0 must enter HIBERN8.
 - The MODULE updates the INLINE-SET from the INLINE-CR registry when all of its TX-LANEs and RX-LANEs enter HIBERN8.
- LINK_CFG_RETAIN_STATUS shall be used to indicate that attribute configuration is retained. A Downstream Port may use LINK_CFG_RETAIN_STATUS at the start of RRC to determine whether it is necessary to perform further configuration. If both local and remote LINK_CFG_RETAIN_STATUS attributes are set at the beginning of RRC then further configuration is not required.
- LINK_CFG_RETAIN_STATUS may be set in the following scenarios:
 - In Configuration.Update when LINK_CFG_RETAIN_CAPABILITY is set.
 - Via implementation specific means when both ends of a LINK have been pre-configured.

8.2.2. Attributes

The following Capability and Configuration Attributes are defined. These must be implemented by both Downstream and Upstream Ports. These registers on the RRAP target must be accessed through the RRAP protocol.

Table 8-2 lists the Capability Attributes. All undefined register fields are reserved.

Table 8-2: M-PCIe Capability Attributes

Attribute	Attribute ID	Description	Range
LINK_HSGEAR_Capability	0x00	Specifies the supported HS_GEARS	Bit 0 = Reserved Bit 1 = High-Speed GEAR 1 SUPPORTED 0: High-Speed GEAR 1 is not supported. 1: High-Speed GEAR 1 is supported. Bit 2 = High Speed GEAR 2 SUPPORTED 0: High-Speed GEAR 2 is not supported. 1: High-Speed GEAR 2 is supported. Bit 3 = High-Speed GEAR 3 SUPPORTED 0: High-Speed GEAR 3 is not supported. 1: High-Speed GEAR 3 is supported. Bits 4-7 = Reserved
TX-LANE_WIDTH_SUPPORTED	0x01	Specifies the supported TX_LANE Width	Bit 0 = x1 TX-LANE SUPPORTED 0: x1 TX-LANE is not supported. 1: x1 TX-LANE is supported. This bit must be hardwired to 1b. Bit 1 = x2 TX-LANE SUPPORTED 0: x2 TX-LANE is not supported. 1: x2 TX-LANE is supported. Bit 2 = x4 TX-LANE SUPPORTED 0: x4 TX-LANE is not supported. 1: x4 TX-LANE is supported. Bit 3 = x8 TX-LANE SUPPORTED 0: x8 TX-LANE is not supported. 1: x8 TX-LANE is supported. Bit 4 = x12 TX-LANE SUPPORTED 0: x12 TX-LANE is not supported. 1: x12 TX-LANE is supported. Bit 5 = x16 TX-LANE SUPPORTED 0: x16 TX-LANE is not supported. 1: x16 TX-LANE is supported. Bit 6 = x32 TX-LANE SUPPORTED 0: x32 TX-LANE is not supported. 1: x32 TX-LANE is supported. Bit 7 = Reserved

Attribute	Attribute ID	Description	Range
RX-LANE_WIDTH_SUPPORTED	0x02	Specifies the supported RX_LANE Width	Bit 0 = x1 RX-LANE SUPPORTED 0: x1 RX-LANE is not supported. 1: x1 RX-LANE is supported. This bit must be hardwired to 1b. Bit 1 = x2 RX-LANE SUPPORTED 0: x2 RX-LANE is not supported. 1: x2 RX-LANE is supported. Bit 2 = x4 RX-LANE SUPPORTED 0: x4 RX-LANE is not supported. 1: x4 RX-LANE is supported. Bit 3 = x8 RX-LANE SUPPORTED 0: x8 RX-LANE is not supported. 1: x8 RX-LANE is supported. Bit 4 = x12 RX-LANE SUPPORTED 0: x12 RX-LANE is not supported. 1: x12 RX-LANE is supported. Bit 5 = x16 RX-LANE SUPPORTED 0: x16 RX-LANE is not supported. 1: x16 RX-LANE is supported. Bit 6 = x32 RX-LANE SUPPORTED 0: x32 RX-LANE is not supported. 1: x32 RX-LANE is supported. Bit 7 = Reserved
2K PPM Required	0x03	Specifies if +/-2K PPM clock compensation is required	Bit 0 = 0, +/-2K PPM Clock Compensation is not required Bit 0 = 1, +/-2K PPM Clock Compensation is required
Asymmetric_LINK_Capability		Indicates if Asymmetric SUB-LINK Width is supported.	Bit 1, when clear indicates that Asymmetric SUB-LINK Widths must not be supported. Bit 1, when set indicates that Asymmetric SUB-LINK Width must be supported. When Asymmetric SUB-LINK Width is supported, all possible combinations as advertised in the Lane Width capability attributes for transmitter and receiver must be supported.
LINK_CFG_RETAIN_CAPABILITY		Indicates support for Retaining M-PHY parameters	Bit 2, When set indicates that the LINK supports retaining M-PHY configuration registers through D3cold.
Reserved		Reserved	Bits 3-4 = Reserved.

Comment [VS325]: Mpcie_28August2014 Errata B15

Comment [VS326]: B15

Comment [VS327]: B15

Attribute	Attribute ID	Description	Range
Rate_Series_Capability		Indicates the supported Rate series.	Bit 5, when set indicates support for Rate A. Bit 6, when set indicates support for Rate B.
Hardware Autonomous_RATE_Series_Re-Configuration_Capability		Indicates support for dynamic RATE series re-configuration.	Bit 7, when clear indicates no support for hardware autonomous RATE series Re-Configuration. Rate series cannot be changed dynamically through Recovery. This bit does not impact initial RATE series change in Configuration.Start. Bit 7, when set indicates support for dynamic RATE series Re-Configuration.
C_REFCLK_RX_HS_G1_SYNC_LENGTH_CAPABILITY	0x04	Common Reference Clock HIGH Speed Gear 1 Synchronization pattern length in Symbol times	The Range for this field is same as the RX_HS_G1_SYNC_LENGTH_Capability field in M-RX Capability Attributes.
C_REFCLK_RX_HS_G2_SYNC_LENGTH_CAPABILITY	0x05	Common Reference Clock HIGH Speed Gear 2 Synchronization pattern length in Symbol times	The Range for this field is same as the RX_HS_G2_SYNC_LENGTH_Capability field in M-RX Capability Attributes.
C_REFCLK_RX_HS_G3_SYNC_LENGTH_CAPABILITY	0x06	Common Reference Clock HIGH Speed Gear 3 Synchronization pattern length in Symbol times	The Range for this field is same as the RX_HS_G3_SYNC_LENGTH_Capability field in M-RX Capability Attributes.
NC_REFCLK_RX_HS_G1_SYNC_LENGTH_CAPABILITY	0x07	Non-Common Reference Clock HIGH Speed Gear 1 Synchronization pattern length in Symbol times.	The Range for this field is same as the RX_HS_G1_SYNC_LENGTH_Capability field in M-RX Capability Attributes.
NC_REFCLK_RX_HS_G2_SYNC_LENGTH_CAPABILITY	0x08	Non-Common Reference Clock HIGH Speed Gear 2 Synchronization pattern length in Symbol times	The Range for this field is same as the RX_HS_G2_SYNC_LENGTH_Capability field in M-RX Capability Attributes.

Attribute	Attribute ID	Description	Range
NC_REFCLK_RX_HS_G3_SYNC_LENGTH_CAPABILITY	0x09	Non-Common Reference Clock HIGH Speed Gear 3 Synchronization pattern length in Symbol times	The Range for this field is same as the RX_HS_G3_SYNC_LENGTH_Capability field in M-RX Capability Attributes.
LINK_RX_HS_G1_PREPARE_LENGTH_CAP ABILITY	0x0A	HIGH Speed Gear 1 Prepare Length multiplier.	The Range for this field is same as the RX_HS_G1_PREPARE_LENGTH_Capability field in M-RX Capability Attributes.
LINK_RX_HS_G2_PREPARE_LENGTH_CAPABILITY	0x0B	HIGH Speed Gear 2 Prepare Length multiplier	The Range for this field is same as the RX_HS_G2_PREPARE_LENGTH_Capability field in M-RX Capability Attributes.
LINK_RX_HS_G3_PREPARE_LENGTH_CAPABILITY	0x0C	HIGH Speed Gear 3 Prepare Length multiplier	The Range for this field is same as the RX_HS_G3_PREPARE_LENGTH_Capability field in M-RX Capability Attributes.
LINK_RX_PWM_BURST_CLOSURE_LENGTH_CAPABILITY	0x0D	Specifies minimum burst closure time (in SI) necessary to guarantee complete data processing on the RX path (PHY and protocol)	~~The Range for this field is same as the RX_PWM_Burst_Closure_Length_Capability field in M-RX Capability Attributes.~~ 0..31.
LINK_HIBERN8_TIME_CAPABILITY	0x0E	Specifies minimum time (in 1 µs steps) required for Local LINK to remain in HIBERN8 state.	Permitted values are from 1 to 100 (1 µs to 100 µs). The maximum value for this field is 100 (100 µs).
LINK_MIN_ACTIVATETIME_CAPABILITY	0x0F	Specifies minimum activate time required for Local LINK in 1 µs steps.	Permitted values are from 1 to 100 (1 µs to 100 µs). The maximum value for this field is 100 (100 µs).
M-PCIe Vendor Specific Registers Upper Address	0xE9	Specifies the upper Address where the M-PCIe Vendor Specific Registers are located.	> 0x22, cannot be located 0x24. ≤ 0x3F

Comment [VS328]: Mpcie_28August2014 Errata C1

Attribute	Attribute ID	Description	Range
M-PCIe Vendor Specific Operational Register Lower Address.	0xEA	Specifies the starting Lower Address for the M-PCIE Vendor Specific Operation Registers.	0x00 to 0xFF
M-PCIe Vendor Specific Test Register Lower Address.	0xEB	Specifies the starting Lower Address for the M-PCIE Vendor Specific Test Registers.	0x00 to 0xFF
M-PCIe Vendor Specific Phy Register Lower Address	0xEC	Specifies the starting Lower Address for the M-PCIE Vendor Specific Phy Registers.	0x00 to 0xFF
M-PCIe Operational Register Vendor ID Low	0xED	Low 8 bits of the PCISIG Vendor ID that represents the organization defining the Vendor specific registers located at M-PCIe Vendor Specific Operational Register Lower Address.	0..255
M-PCIe Operational Register Vendor ID High	0xEE	High 8 bits of the PCISIG Vendor ID that represents the organization defining the Vendor specific registers located at M-PCIe Vendor Specific Operational Register Lower Address.	0..255

Attribute	Attribute ID	Description	Range
M-PCIe Test Register Vendor ID Low	0xEF	Low 8 bits of the PCISIG Vendor ID that represents the organization defining the Vendor specific registers located at M-PCIe Vendor Specific Test Register Lower Address.	0..255
M-PCIe Test Register Vendor ID High	0xF0	High 8 bits of the PCISIG Vendor ID that represents the organization defining the Vendor specific registers located at M-PCIe Vendor Specific Test Register Lower Address.	0..255
M-PCIe Phy Register Vendor ID Low	0xF1	Low 8 bits of the PCISIG (MIPI?) Vendor ID that represents the organization defining the Vendor specific registers located at M-PCIe Vendor Specific Phy Register Lower Address.	0..255
M-PCIe Phy Register Vendor ID High	0xF2	High 8 bits of the PCISIG (MIPI?) Vendor ID that represents the organization defining the Vendor specific registers located at M-PCIe Vendor Specific Phy Register Lower Address.	0..255

Attribute	Attribute ID	Description	Range
M-PCIe Phy Register Vendor ID Issuer	0xF3	Indicates the organization that issued the ID contained in M-PCIe Phy Register Vendor ID Values are: 0 Nobody – Vendor ID is not meaningful. 1 Vendor ID issued by PCISIG. 2 Manufacturer ID issued by the MIPI Alliance. All other values are Reserved.	0..255
M-PCIe Operational Register Capabilities1..4	0xF4-0xF7	Reserved register, reads 0 unless otherwise defined by the Vendor identified by M-PCIe Operational Register Vendor ID.	Any
M-PCIe Test Register Capabilities1..4	0xF8-0xFB	Reserved register, reads 0 unless otherwise defined by Vendor identified by M-PCIe Test Register Vendor ID.	Any
M-PCIe Phy Register Capabilities1..4	0xFC-0xFF	Reserved register, reads 0 unless otherwise defined by the Vendor identified by M-PCIe Phy Register Vendor ID.	Any

Comment [VS329]: Fix last sentence in ECN/Errata.

If M-PCIe Operational Register Vendor ID is 0000h, there are no registers documented in the M-PCIe Vendor Specific Operational Register Lower Address and M-PCIe Phy Operational Capabilities 1 through 4 must be 00h.

If M-PCIe Test Register Vendor ID is 0000h, there are no registers documented in the M-PCIe Vendor Specific Test Register Lower Address and M-PCIe Phy Test Capabilities 1 through 4 must be 00h.

If M-PCIe Phy Register Vendor ID Issuer is 0, there are no registers documented in the M-PCIe Vendor Specific Phy Register Lower Address and M-PCIe Phy Register Capabilities 1 through 4 must be 00h.

Table 8-3 lists the M-PCIe Configuration Attributes. All undefined register fields are reserved. M-PCIe Configuration Attributes are reset on LINE-RESET and Fundamental Reset unless explicitly noted in this specification.

Comment [VS330]: Mpcie_28August2014 Errata C5

Table 8-3: M-PCIe Configuration Attributes

Attribute	Attribute ID	Description	Range
LINK_HSGEAR	0x10	Specifies the HS_GEAR the LINK must operate at. Only one HS-GEAR must be selected.	Bit 0 = Reserved Bit 1 = High-Speed GEAR 1 Enable. 0: High-Speed GEAR 1 is not enabled. 1: High-Speed GEAR 1 is enabled. Bit 2 = High Speed GEAR 2 Enable. 0: High-Speed GEAR 2 is not enabled. 1: High-Speed GEAR 2 is enabled. Bit 3 = High-Speed GEAR 3 Enable. 0: High-Speed ~~GEAR 2~~GEAR 3 is not enabled. 1: High-Speed ~~GEAR 2~~GEAR 3 is enabled. Bits 4-7 = Reserved.
TX-LANE_WIDTH	0x11	Configured TX_LANE width, only one LANE width must be selected.	Bit 0 = x1 TX-LANE 0: x1 TX-LANE is not enabled. 1: x1 TX-LANE is enabled. Bit 1 = x2 TX-LANEs 0: x2 TX-LANE is not enabled. 1: x2 TX-LANE is enabled. Bit 2 = x4 TX-LANEs 0: x4 TX-LANE is not enabled. 1: x4 TX-LANE is enabled. Bit 3 = x8 TX-LANEs 0: x8 TX-LANE is not enabled. 1: x8 TX-LANE is enabled. Bit 4 = x12 TX-LANEs 0: x12 TX-LANE is not enabled. 1: x12 TX-LANE is enabled. Bit 5 = x16 TX-LANEs 0: x16 TX-LANE is not enabled. 1: x16 TX-LANE is enabled. Bit 6 = x32 TX-LANEs 0: x32 TX-LANE is not enabled. 1: x32 TX-LANE is enabled. Bit 7 = Reserved

Comment [VS331]: Mpcie_28August2014 Errata C2

Attribute	Attribute ID	Description	Range
RX-LANE_WIDTH	0x12	Configured RX-LANE width, only one LANE width must be selected.	Bit 0 = x1 RX-LANE 0: x1 TRX-LANE is not enabled. 1: x1 RX-LANE is enabled. Bit 1 = x2 RX-LANEs 0: x2 RX-LANE is not enabled. 1: x2 RX-LANE is enabled. Bit 2 = x4 RX-LANEs 0: x4 RX-LANE is not enabled. 1: x4 RX-LANE is enabled. Bit 3 = x8 RX-LANEs 0: x8 RX-LANE is not enabled. 1: x8 RX-LANE is enabled. Bit 4 = x12 RX-LANEs 0: x12 RX-LANE is not enabled. 1: x12 RX-LANE is enabled. Bit 5 = x16 RX-LANEs 0: x16 RX-LANE is not enabled. 1: x16 RX-LANE is enabled. Bit 6 = x32 RX-LANEs 0: x32 RX-LANE is not enabled. 1: x32 RX-LANE is enabled. Bit 7 = Reserved.
2K_PPM_DISABLED	0x13	When set indicates that 2K PPM is not used, SKP Ordered Set frequency can be selected accordingly.	Bit 0 = 0, 2K PPM is selected Bit 0 = 1, 2K PPM is not selected.
Hardware_Autonomous_RATE Series Re-Configuration Enable		When set enable RATE series to be autonomously changed using Dynamic LINK Re-Configuration.	Bit 7 = 0, Disabled, this is the default value for this field. Bit 7 = 1, Enabled.
Reserved		Reserved.	Bits 1-6 = Reserved.

Attribute	Attribute ID	Description	Range
C_REFCLK_TX_HS_G1_SYNC_LENGTH_CONTROL	0x14	HIGH Speed Gear 1 Synchronization common reference clock pattern length in Symbol times.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
C_REFCLK_TX_HS_G2_SYNC_LENGTH_CONTROL	0x15	HIGH Speed Gear 2 Synchronization common reference clock pattern length in Symbol times.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
C_REFCLK_TX_HS_G3_SYNC_LENGTH_CONTROL	0x16	HIGH Speed Gear 3 Synchronization common reference clock pattern length in Symbol times.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
NC_REFCLK_TX_HS_G1_SYNC_LENGTH_CONTROL	0x17	HIGH Speed Gear 1 Synchronization non-common reference clock pattern length in Symbol times.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
NC_REFCLK_TX_HS_G2_SYNC_LENGTH_CONTROL	0x18	HIGH Speed Gear 2 Synchronization non-common reference clock pattern length in Symbol times.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
NC_REFCLK_TX_HS_G3_SYNC_LENGTH_CONTROL	0x19	HIGH Speed Gear 3 Synchronization non-common reference clock pattern length in Symbol times	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
LINK_TX_HS_G1_PREPARE_LENGTH_CONTROL	0x1A	HIGH Speed Gear 1 Prepare Length multiplier.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.

Attribute	Attribute ID	Description	Range
LINK_TX_HS_G2_PREPARE_LENGTH_CONTROL	0x1B	HIGH Speed Gear 2 Prepare Length multiplier.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
LINK_TX_HS_G3_PREPARE_LENGTH_CONTROL	0x1C	HIGH Speed Gear 3 Prepare Length multiplier.	Configured during LINK Init and Discovery. Configured to match the remote LINK partner's RX capability for the corresponding Gear.
LINK_TX_ HIBERN8_TIME_	0x1D	Specifies minimum time (in 1 µs steps) required for transmitter to remain in HIBERN8 state.	Configured during LINK Init and Discovery to be greater of the remote LINK partner's LINK_HIBERN8_TIME_CAPABILITY and the local LINK_HIBERN8_TIME_CAPABILITY. Permitted values are from 1to 100 (1 µs to 100 µs). The maximum value for this field is 100 (100 µs). This attribute overrides the M-PHY THIBERN8 value, minimum THIBERN8 is equal to the value defined in this Attribute field.
LINK_TX_MIN_ACTIVATETIME	0x1E	Specifies minimum activate time required for transmitter in 1 µs steps.	Configured during LINK Init and Discovery to match the remote LINK partner's LINK_MIN_ACTIVATETIME_CAPABILITY. Permitted values are from 1to 100 (1 µs to 100 µs). The maximum value for this field is 100 (100 µs). This attribute overrides the M-PHY $T_{ACTIVATE}$ value, minimum $T_{ACTIVATE}$ is equal to the value defined in this Attribute field.
LINK_MIN_SAVE_CONFIG_TIME	0x1F	Specifies the greater of the local ~~remote~~ MPHY TX_Min_SAVE_Config_Time_Capability and the remote RX_Min_SAVE_Config_Time_Capability.	Configured during LINK Init and Discovery to be greater of the local~~remote~~ MPHY TX_Min_SAVE_Config_Time_Capability and the remote RX_Min_SAVE_Config_Time_Capability in increments of 40ns.

Comment [VS332]: Mpcie_28August2014 Errata B5

Comment [VS333]: B5

Comment [VS334]: B5

Table 8-4 lists the M-PCIe Status Attributes. All undefined register fields are reserved. M-PCIe Status Attributes are reset on LINE-RESET and Fundamental Reset unless explicitly noted in this specification.

Comment [VS335]: Mpcie_28August2014 Errata C5

Table 8-4: M-PCIe Status Attributes

Attribute	Attribute ID	Description
LINK_CFG_RETAIN_STATUS	0x20	Indicates M-PHY and M-PCIe Configuration attributes are retained through D3cold. Bit 0 = 0, M-PCIe Configuration attributes are not retained through D3cold. Bit 0 = 1, M-PCIe Configuration attributes are retained through D3cold.

8.2.3. Remote Register Access Protocol (RRAP):

The RRAP consists of the following packet types:

- Write Command
- Write Response
- Read Command
- Read Response

An RRAP Master must be capable of issuing Command packets while an RRAP Target must issue response packets. The Downstream port acts as the master and the Upstream port acts as the target when not in TEST mode. The requests to read and write parameters are generated by the Downstream port and responses are returned by the Upstream port. There must only be one outstanding RRAP request at any time.

The 4-Byte format of each command and response packet is shown below.

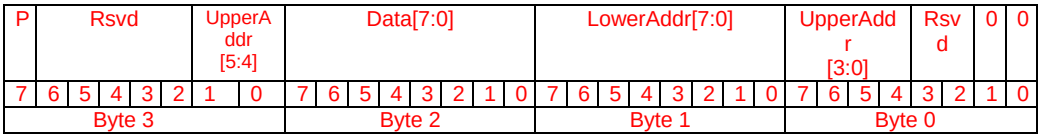


Figure 8-3: RRAP Write Command Format

P	Rsvd	A	Rsvd																							0	1				
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Byte 3								Byte 2								Byte 1								Byte 0							

Figure 8-4: RRAP Write Response Format

P	Rsvd							UpperAddr [5:4]		Rsvd							LowerAddr[7:0]								UpperAddr [3:0]				Rsvd		1	0
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
Byte 3								Byte 2								Byte 1								Byte 0								

Figure 8-5: RRAP Read Command Format

P	Rsvd	A	Rsvd							Data[7:0]							Rsvd													1	1
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Byte 3								Byte 2								Byte 1								Byte 0							

Figure 8-6: RRAP Read Response Format

The fields in the above packet formats are further defined below:

- "LowerAddr" and "UpperAddr" values are set based on the address map defined in Table 8-6.
- The "Rsvd" fields must be set to 0 by the transmitter and ignored by the receiver.
- The "P" field must be set by the transmitter as an odd-parity bit for the entire packet.
- The "A" field in the response packet represents "Abort" response from the RRAP target. The RRAP rules are:
- When in PWM-BURST mode and not transmitting an RRAP packet, the M-TX MODULEs in SUB-LINKs must transmit the PAD symbol per [M-PHY]
- A Downstream Port must support both RRAP Master and RRAP Target functionalities, but in a mutually exclusive manner. RRAP Target functionality is enabled for a Downstream Port when in test mode. The Downstream Port's RRAP Target functionality must be enabled when it receives an RRAP command. When the Downstream Port's RRAP Target functionality is enabled, the Downstream Port's RRAP Master functionality must be disabled and the Downstream Port must terminate all RRAP commands issued by it as an RRAP Master.
- An Upstream Port must only support RRAP Target functionality. Test Equipment may be designed as a RRAP Master with either the Downstream Port or the Upstream Port as the Device Under Test (DUT).
- A Downstream Port serving as a RRAP Master must only send commands and receive responses on LANE PAIR0.
- A Downstream Port or an Upstream Port serving as an RRAP Target must support receiving commands and returning responses on LANE PAIR0.

- If a Master has issued a Command, the Master must not retry the Command or issue another Command until either the Target has provided a Response, or after tRRAPInitiatorResponse has elapsed.
- If the master receives a response packet with "Abort" bit set then handling of the response packet and subsequent responses are implementation specific.
- If the master receives a response packet with 8b/10b error or parity error then it must drop the response packet. The master is permitted to retry the configuration request after tRRAPInitiatorResponse has elapsed.
- After a RRAP timeout, the master is permitted to retry the configuration request.
- If the master receives an unexpected response, it must restart the configuration process from the beginning.
 - An unexpected response is when the lower two bits of the response header are not as expected.
- A Target must map the attributes of all implemented LANE PAIRs to RRAP addresses as defined in the address map table.
- A Target must not send a response until the RRAP Command is fully received with a valid parity.
- Upon receiving a Write or a Read Command packet, a Target must transmit the corresponding response packet within tRRAPTargetResponse.
- The target must check for both 8b/10b errors and parity error of the received command packet and drop all packets that have 8b/10b errors or parity error.
- The "Abort" field is set to '1' in the response packet by the RRAP target, if the target cannot complete the command (e.g., the address in the command maps to a non-existent M-TX module or non-existent M-RX module in the RRAP target's SUB-LINKs).

8.2.3.1. RRAP Timing Parameters

Table 8-5 defines the timing parameters relevant to the RRAP.

Table 8-5: RRAP Timing Parameters

Name	Description	Min	Max	Units
tRRAPTargetResponse	Time between the receipt of a RRAP command and the transmission of the response by a Target		50	ms
tRRAPInitiatorResponse	Time between the transmission of a RRAP command and the reception of the response by an Initiator		60	ms

8.2.3.2. RRAP Address Maps

There are two address maps defined, one Protocol-Specific RRAP Address Map, and one Test Address Map.

Table 8-6 below lists the Protocol-Specific RRAP address map.

Table 8-6: RRAP M-PCIe Address Map

UpperAddr[5:0] LowerAddr[7:0]	Register Name	Description
0x00-0x1F 0x00-0xFF	As defined in [M-PHY], for the LowerAddr[7:0]. UpperAddr[5:0] defines the LANE PAIR number <n>. 0x00: LANE PAIR0. 0x01: LANE PAIR1. … 0x1F: LANE PAIR31.	Capability, Configuration and Status attributes for PAIR<n>. The Target must provide a Write Response only after the Write Command is completed.
0x20 0x00-0xFF	Broadcast Write	UpperAddr of 0x20 is used as an indication of broadcast command. Any broadcast write is applied to all PAIRs. . Broadcast writes to the RX-LANEs or TX-LANEs that don't exist must be ignored by the RRAP Target, and it must generate normal response packet as long as the RX-LANEs and TX-LANEs that exist are updated. Read of this register is undefined.
0x21 0x00-0xFF	M-PCIe Capability, Configuration and Status Attributes	LowerAddr[7:0] represents the Attribute ID of the corresponding M-PCIe Capability and Configuration Attributes.
0x24 0xFF	M-PHY TEST MODE	This register is used to place the port into test mode. Bit[0]: "0b": Normal Operation "1b": Test Mode Reset Default: Normal Operation Bits [7:1] : Reserved. Once the target is in Test Mode the state can only be exited with a LINE-RESET.

Table 8-7 lists the RRAP address map for Test. To control the Transmit Compliance Mode, the corresponding Transmit Compliance Mode register is associated with a set of (up to) eight TX-LANEs, as shown for each register in Table 8-7. For each bit, a value of 0b indicates Normal Operation, and a value of 1b indicates TX Compliance Mode. The reset default for all bits must be 0b. When set to TX Compliance Mode the TX-LANE(s) must continuously transmit the CRPAT compliance pattern as defined in Appendix I. Only the indicated lanes must go into TX compliance

mode. It is permitted that Multiple TX-LANEs be placed into TX compliance mode at the same time, and they must be configured to operate at the same RATE and Gear. The results of writes to these registers are undefined unless the M-PHY Test Mode register is also set to Test Mode.

To control the M-PHY Analog Loopback Mode, the corresponding Analog Loopback Mode register is associated with a set of (up to) eight RX-LANE(s), as shown for each register in Table 8-7. For each bit, a value of 0b indicates Normal Operation, and a value of 1b indicates the target RX-LANE must be placed into M-PHY Analog Loopback mode. The reset default for all bits must be 0b. The results of writes to these registers are undefined unless the M-PHY Test Mode register is also set to Test Mode. Once the target lane is in M-PHY analog Loopback mode it must transmit all received data with the RX clock on the Loopback partner TX Lane, which is set in the Loopback Partner LANE register. This functionality is analogous to the mode defined in MIPI Alliance Specification for M-PHY version 2.0 Annex B.2.2 as "Analog Loopback" mode. Specifically a "Synchronous Loopback" mechanism is employed as described in Annex B.2.2.1 in which the recovered clock from M-RX is used to retransmit the data on the M-TX. Only one lane is permitted to be placed in M-PHY Analog Loopback mode at a time. It is permitted to program TX-LANEs other than the Loopback partner into TX Compliance mode. The behavior is undefined if register writes attempt to place more than one lane in Analog Loopback Mode.

Each of the M-PHY Burst Error Count Mode registers is associated with a set of (up to) eight RX-LANE(s), as shown for each register in Table 8-7. For each bit, a value of 0b indicates Normal Operation, and a value of 1b indicates the target RX-LANE must be placed into Burst Error Count Mode. The reset default for all bits must be 0b. Only one lane is permitted to be placed into M-PHY Burst Error Count Mode at a time. The results of writes to this register are undefined unless the M-PHY Test Mode register is also set to Test Mode.

The RX_BURST_COUNT and RX_ERR_COUNT registers as defined in Table 8-7 must not rollover and remain unchanged when they reach their maximum values.

Only the Test programmer is permitted to issue LINE-RESET. When any one RX-LANE receives LINE-RESET, an internal reset must be issued to all receiver and transmitter LANEs using implementation specific means. Upon receiving LINE-RESET the LTSSM (see Section 8.4.9) must enter Detect state.

The Test Registers listed in the Table 8-7 must be reset on LINE-RESET unless explicitly noted in this specification. All undefined register fields must be reserved.

Table 8-7: Test Address Map

UpperAddr[5:0] LowerAddr[7:0]	Register Name	Description
0x2E 0x00-0x1F	LOOPBACK_PARTNER_LANE LowerAddr[4:0] defines the LANE number <m>	Read Only The TX-LANE number that will transmit Loopback data when the target RX-LANE(m) is in analog Loopback mode. The Loopback partner TX-Lane for RX-Lane 0 must be TX-Lane 0. Any TX lane may be used for other RX lanes. Multiple Rx lanes are allowed to have the same TX-Lane for a Loopback partner.
0x2E 0x20	TRANSMIT COMPLIANCE MODE LANES [7:0]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated TX-LANE is in ~~Used to place TX-LANEs into~~TX compliance mode: Bit [7] controls Lane 7 … Bit [1] controls Lane 1 Bit [0] controls Lane 0
0x2E 0x21	TRANSMIT COMPLIANCE MODE LANES [15:8]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated TX-LANE is in TX compliance mode: Bit [7] controls Lane 15 … Bit [1] controls Lane 9 Bit [0] controls Lane 8
0x2E 0x22	TRANSMIT COMPLIANCE MODE LANES [23:16]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated TX-LANE is in TX compliance mode: Bit [7] controls Lane 23 … Bit [1] controls Lane 17 Bit [0] controls Lane 16

Comment [VS336]: Mpcie_28August2014 Errata B22

Comment [VS337]: Mpcie_28August2014 Errata B22

Comment [VS338]: Mpcie_28August2014 Errata B22

Comment [VS339]: Mpcie_28August2014 Errata B22

Comment [VS340]: Mpcie_28August2014 Errata B22

Comment [VS341]: Mpcie_28August2014 Errata B22

UpperAddr[5:0] LowerAddr[7:0]	Register Name	Description
0x2E 0x23	TRANSMIT COMPLIANCE MODE LANES [31:24]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated TX-LANE is in TX compliance mode: Bit [7] controls Lane 31 … Bit [1] controls Lane 25 Bit [0] controls Lane 24
0x2E 0x30	M-PHY ANALOG LOOPBACK MODE LANES [7:0]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE and its Loopback Partner TX-LANE are in ~~Used to place an RX-LANE into~~ Analog Loopback mode: Bit [7] controls Lane 7 … Bit [1] controls Lane 1 Bit [0] controls Lane 0
0x2E 0x31	M-PHY ANALOG LOOPBACK MODE LANES [15:8]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE and its Loopback Partner TX-LANE are in Analog Loopback mode: Bit [7] controls Lane 15 … Bit [1] controls Lane 9 Bit [0] controls Lane 8
0x2E 0x32	M-PHY ANALOG LOOPBACK MODE LANES [23:16]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE and its Loopback Partner TX-LANE are in Analog Loopback mode: Bit [7] controls Lane 23 … Bit [1] controls Lane 17 Bit [0] controls Lane 16

Comment [VS342]: Mpcie_28August2014 Errata B22

Comment [VS343]: Mpcie_28August2014 Errata B22

Comment [VS344]: Mpcie_28August2014 Errata B22

Comment [VS345]: Mpcie_28August2014 Errata B22

Comment [VS346]: Mpcie_28August2014 Errata B22

UpperAddr[5:0] LowerAddr[7:0]	Register Name	Description
0x2E 0x33	M-PHY ANALOG LOOPBACK MODE LANES [31:24]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE and its Loopback Partner TX-LANE are in Analog Loopback mode: Bit [7] controls Lane 31 … Bit [1] controls Lane 25 Bit [0] controls Lane 24
0x2E 0x40	M-PHY BURST ERROR COUNT MODE [7:0]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE is in ~~Used to place RX-LANEs into~~ M-PHY Burst Error Count compliance mode: Bit [7] controls Lane 7 … Bit [1] controls Lane 1 Bit [0] controls Lane 0
0x2E 0x41	M-PHY BURST ERROR COUNT MODE [15:8]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE is in M-PHY Burst Error Count compliance mode: Bit [7] controls Lane 15 … Bit [1] controls Lane 9 Bit [0] controls Lane 8
0x2E 0x42	M-PHY BURST ERROR COUNT MODE [23:16]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE is in M-PHY Burst Error Count compliance mode: Bit [7] controls Lane 23 … Bit [1] controls Lane 17 Bit [0] controls Lane 16

Comment [VS347]: Mpcie_28August2014 Errata B22

Comment [VS348]: Mpcie_28August2014 Errata B22

Comment [VS349]: Mpcie_28August2014 Errata B22

Comment [VS350]: Mpcie_28August2014 Errata B22

Comment [VS351]: Mpcie_28August2014 Errata B22

Comment [VS352]: Mpcie_28August2014 Errata B22

Comment [VS353]: Mpcie_28August2014 Errata B22

UpperAddr[5:0] LowerAddr[7:0]	Register Name	Description
0x2E 0x43	M-PHY BURST ERROR COUNT MODE [31:24]	Read/Write if the lane is implemented, Read Only otherwise. Default 0. If Set, the indicated RX-LANE is in M-PHY Burst Error Count compliance mode: Bit [7] controls Lane 31 … Bit [1] controls Lane 25 Bit [0] controls Lane 24
0x2E 0x44	RX_BURST_COUNT_0	Read Only Default 0. ~~These Registers~~The RX_BURST_COUNT registers count the number of bursts by incrementing on each HS-BURST to STALL transition for a lane that is in M-PHY Burst Error Count Mode. RX_BURST_COUNT_0 is the LSB, forming bits[7:0] of the counter, RX_BURST_COUNT_1 forms bits[15:8], RX_BURST_COUNT_2 forms bits[23:16] and RX_BURST_COUNT_3 is the MSB, forming bits[31:24] of the counter. This is a 32 bit saturating counter. When it reaches 0xFFFFFFFF, it remains at that value until reset. ~~Reset Default: 0~~ These registers are not reset by LINE-RESET. The registers are reset to zero when RX_COUNT_RESET is set.
0x2E 0x45	RX_BURST_COUNT_1	
0x2E 0x46	RX_BURST_COUNT_2	
0x2E 0x47	RX_BURST_COUNT_3	
0x2E 0x48	RX_ERR_COUNT_0	Read Only, default 0. The RX_ERR_COUNT registers ~~These Registers~~count the number of errors by incrementing on each HS-BURST to STALL transition for which a MARKER0 was not detected. RX_ERR_COUNT_0 is the LSB, forming bits [7:0] of the counter, RX_ERR_COUNT_1 forms bits[15:8], RX_ERR_COUNT_2 forms bits[23:16] and RX_ERR_COUNT_3 is the MSB, forming bits[31:24] of the counter. This is a 32 bit saturating counter. When it reaches 0xFFFFFFFF, it remains at that value until reset. ~~Reset Default: 0~~ These registers are not reset by LINE-RESET. The registers are reset to zero when RX_COUNT_RESET is set.
0x2E 0x49	RX_ERR_COUNT_1	
0x2E 0x4A	RX_ERR_COUNT_2	
0x2E 0x4B	RX_ERR_COUNT_3	

Comment [VS354]: Mpcie_28August2014 Errata B22

Comment [VS355]: Mpcie_28August2014 Errata B22

Comment [VS356]: Mpcie_28August2014 Errata B23_v2

Comment [VS357]: Mpcie_28August2014 Errata B24_v2

UpperAddr[5:0] LowerAddr[7:0]	Register Name	Description
0x2E 0x4C	RX_COUNT_RESET	This register is used to reset the RX_BURST_COUNT and the RX_ERR_COUNT registers It is a write only self-clearing register. Reset Default: 0 Bit [0]: Writing 1'b1 resets the RX_BURST_COUNT and the RX_ERR_COUNT registers Bits [7:1]: Reserved.

Figure 8-7 shows a block diagram of the functional component of a test block implementing the various electrical test modes.

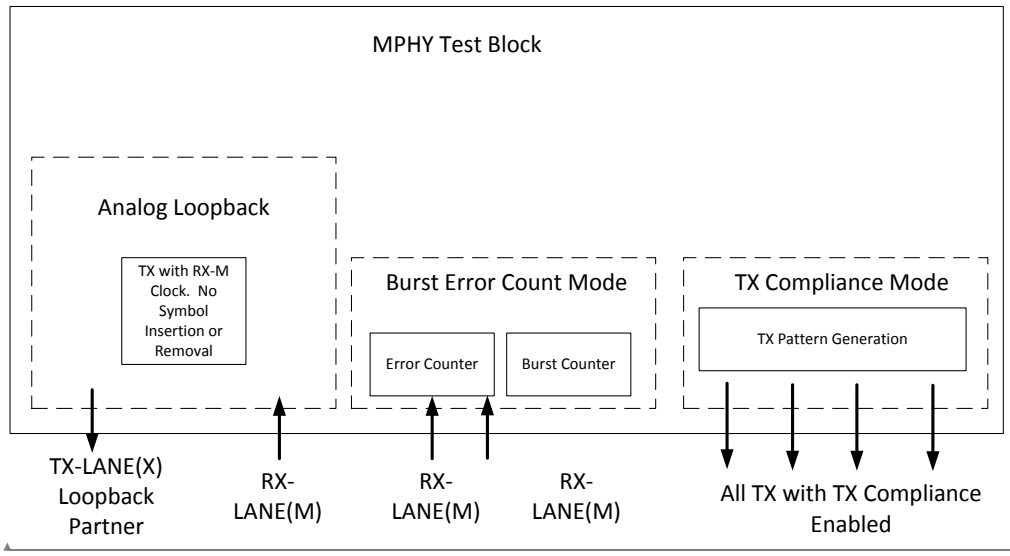
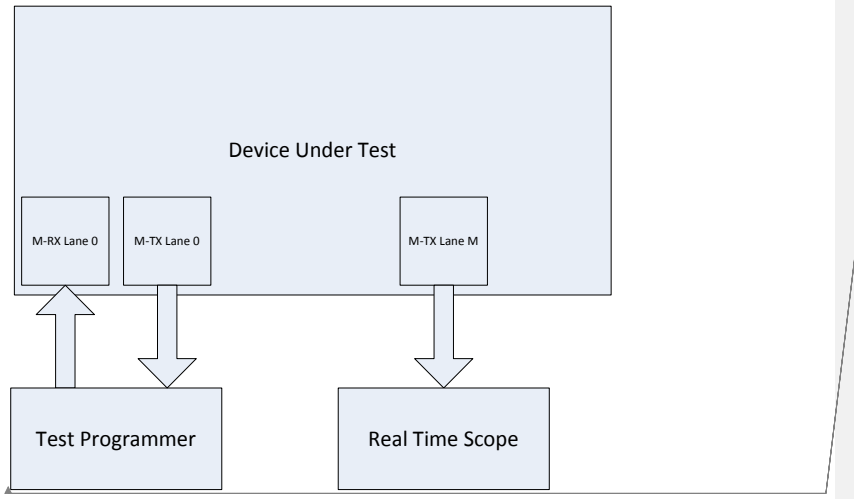


Figure 8-7: M-PHY Test Block

Field Code Changed

Figure 8-8 shows a typical configuration when transmitter testing is performed on TX-LANE(M) using Transmit Compliance Mode.



Field Code Changed

Figure 8-8: Example Transmit Compliance Mode

The following is a typical example sequence for performing the TX test on a component with TX crosstalk present.

- ❑ The Test Programmer configures the DUT to the desired rate, width, gear, etc. using RRAP. Note that the DUT could be configured to test PWM transmissions.
- ❑ The Test programmer sets the M-PHY Test Mode register.
- ❑ The Test programmer writes the four Transmit Compliance Mode Lanes registers to enable transmit compliance mode for TX-LANE(M) and to enable it for all other lanes.
- ❑ The Test programmer terminates the LS-Burst.
- ❑ The DUT detects the end of configuration.
- ❑ After the RRAP register accesses are processed the DUT begins transmitting compliance pattern repetitively on all TX LANEs using the appropriate HS gear.
- ❑ The real time oscilloscope is used to capture a long waveform of transmitted data on TX-LANE(M).
- ❑ The Test programmer issues a LINE-RESET.
- ❑ The process is repeated for any additional lanes to be tested.

Field Code Changed

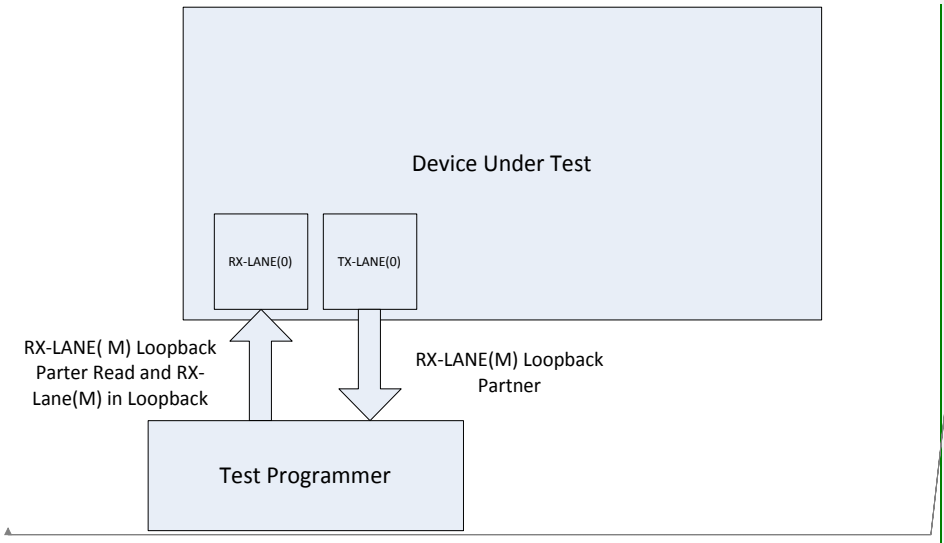
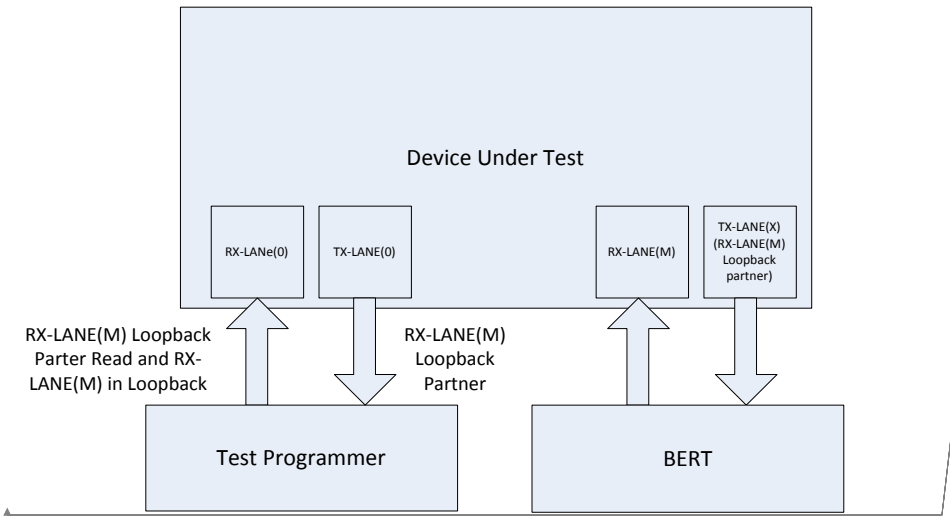


Figure 8-9: Example of a configuration to enable Analog Loopback Mode

Figure 8-9 shows a typical configuration when receiver bit error rate testing is performed on RX-LANE(M) using Analog Loopback Mode.

The following is a typical example sequence for performing the RX test on a component without TX crosstalk present.

- The Test Programmer configures the DUT to the desired rate, width, gear, etc. using RRAP. Note that the DUT could be configured to test PWM transmissions.
- The Test programmer sets the M-PHY Test Mode register.
- The Test programmer writes the four Transmit Compliance Mode Lanes registers to disable transmit compliance mode for all lanes.
- The Test programmer writes the four Analog Loopback Mode registers to enable analog Loopback for the lane M to be tested and to disable it for all other lanes.
- The Test programmer reads Loopback Partner lane for RX-LANE(M) to be tested. The Loopback partner name is referenced as TX-LANE(X).
- The Test programmer terminates the LS-Burst.
- The DUT detects the end of configuration.
- After the RRAP register accesses are processed the DUT got into Analog Loopback Mode on RX-LANE(M) and TX-LANE(X).



Field Code Changed

Figure 8-10 : Example Analog Loopback Mode Configuration

- The BERT transmitter that has been calibrated to provide worst case compliant signaling is connected to RX-LANE(M) and the BERT error counter is connected to TX -LANE(X).
 - Note: If LANE PAIR0 is being tested the PWM Test Programmer is disconnected and the BERT is then connected. A disconnect/reconnect sequence must not cause the DUT to exit analog Loopback mode. Some BERTs may be able to act as the PWM Test Programmer directly.
- The BERT transmits bits with a stressful test pattern and counts errors at its error detector.
- When testing LANE PAIR0, first disconnect the BERT and re-connect Test Programmer before proceeding with the next step.
- The Test Programmer issues a LINE-RESET.
- The process is repeated for any additional lanes to be tested.

8.3. Symbol Encoding, Framing and Scrambling

The MIPI M-PHY uses 8b/10b encoding, this provides Control Symbols that are distinct from Data Symbols.

Table 8-8 defines the Control Symbol mapping.

Table 8-8: M-PCIe Line Encoding

Control Symbols	M-PCIe Encoding	MIPI M-PHY Mapping
K28.5	COM	MARKER0
K28.3	IDL	MARKER1
K28.6	SDP	MARKER2
K23.7	SKP	MARKER3
K27.7	STP	MARKER4
K29.7	END	MARKER5
K30.7	EDB	MARKER6
K28.1	PAD	FILLER
K28.0	Reserved	Reserved
K28.2	Reserved	Reserved
K28.7	Reserved	Reserved
Others	Reserved	Reserved

8.3.1. 8b/10b Decode Rules

The 8b/10b decode rules are the same as defined in Section 4.2.1.1.3 8b/10b Decode Rules with the following exception: The TAIL-OF-BURST sequence intentionally violates the running disparity, and the Physical Layer must not notify the Data Link Layer that the received Symbol is invalid. This is not a Receiver Error, and must not be reported as an error associated with the Port.

8.3.2. Framing and Application of Symbols to LANES

Framing and application of Symbols to LANES is as defined in Section 4.2.1.2, with the following exceptions:

- For xN Links where N is 8 or more, if an END or EDB Symbol is placed in a Lane K, where K does not equal N-1, and is not followed by a STP or SDP Symbol in Lane K+1 (i.e., there is no TLP or DLLP immediately following), then Logical Idle Symbols must be placed in Lanes K+1 to Lane N-1. For example, on a x8 Link, if END or EDB is placed in Lane 3, Logical Idle must be placed in Lanes 4 to 7, when not followed by STP or SDP.
- PAD (MIPI M-PHY FILLER) symbols are not permitted to be inserted within any link layer packet, any transaction layer packet or any ordered set.

8.3.3. Data Scrambling

The scrambling function is the same as defined in Data Scrambling Section 4.2.1.3. The data symbols transmitted in the SYNC phase of MIPI-M-PHY must not be scrambled. It is not required to provide the symbols to be used during SYNC phase – the M-PHY uses internal SYNC data symbols as defined in MIPI M-PHY specification.

8.4. Link Initialization and Training

This Section defines the Physical Layer control process that configures and initializes each Link for normal operation. This Section covers the following features:

- configuring and initializing the Link
- supporting normal packet transfers
- supported state transitions when recovering from Link errors
- Restarting a Port from low power states.

The following features are determined prior to the training process.

- Link width
- Link data rate
- MIPI M-PHY parameters

Training does:

- Establish Link data rate
- Bit lock per Lane
- Symbol lock per Lane
- Lane-to-Lane de-skew within a multi-Lane Link.

8.4.1. Training Sequence (TS) Ordered Sets

Training Sequence Ordered Sets must not be scrambled.

Training Sequence Ordered Sets can be used to provide additional time for the receiver to achieve bit lock and symbol lock when exiting deeper power management state such as L1.

Training Sequence Ordered Sets must be transmitted back-to-back, without any other symbols transmitted between Training Sequence Ordered Sets, except for SKP Ordered Sets.

Reception of Training Sequence Ordered Sets is considered "Consecutive" when all the 16 symbols within the Ordered Set match the immediately preceding Ordered Set.

There are two Training Sequence Ordered Sets: TS1 and TS2. The formats of TS1 and TS2 Ordered Sets are defined in Table 8-9 and Table 8-10 respectively. It is permitted for an implementation to advertise a sub-set of its Capability attributes in TS1 or TS2 Ordered Sets regardless of whether it is performing a bandwidth change request.

Comment [VS358]: Mpcie_28August2014 Errata B9

Table 8-9: TS1 Ordered Set.

Symbol Number	Description
0	COM (K28.5) for Symbol alignment.
1	LINK Bandwidth and GEAR support Bit 0 - LINK Bandwidth Change. 0b: De-assert. When this bit is Clear, bits 2 through 4 indicate that the transmitting Port supports the corresponding data rate, and bits 0 through 1 of Symbol 2 indicate that the transmitting Port supports the corresponding RATE series. At least one High-Speed Gear bit and one RATE series bit must be Set. 1b: Assert. When this bit is Set, bits 2 through 4 indicate that the transmitting port is requesting a change to the corresponding data rate, and bits 0 through 1 of Symbol 2 indicate that the transmitting Port is requesting a change to the corresponding RATE series. At least one High-Speed Gear bit and one RATE series bit must be Set. When multiple GEARs or RATE series are requested, the Downstream Port chooses between them and indicates it in TS2. Bit 1 - Reserved Bit 2 - High-Speed GEAR 1 - 0b: not supported; 1b: supported. Bit 3 - High-Speed GEAR 2 - 0b: not supported; 1b: supported. Bit 4 - High-Speed GEAR 3 - 0b: not supported; 1b: supported. All other bits are Reserved.
2	RATE Series Support Bit 0 - RATE-series A - 0b: not supported; 1b: supported. Bit 1 - RATE-series B - 0b: not supported; 1b: supported. All other bits are Reserved.
3	TX-LANE Number within LINK for Transmitter and RX-LANE Number within LINK for Receiver.

Symbol Number	Description
4	TX SUB-LINK Width. When LINK Bandwidth Change field (bit 0 of Symbol 1 of TS1) of the TS1 Ordered Set is Clear, bits 0 through 6 indicate that the transmitting Port supports the corresponding TX SUB-LINK width. When LINK Bandwidth Change field (bit 0 of Symbol 1 of TS1) of the TS1 Ordered Set is Set, bits 0 through 6 indicate that the transmitting Port is requesting a change to the corresponding TX SUB-LINK Width. When multiple TX SUB-LINK widths are requested, the Downstream Port chooses between them and indicates it in TS2. Bit 0 - x1 TX SUB-LINK Support. Always set to 1. Support for x1 TX SUB-LINK is mandatory. Bit 1 - x2 TX SUB-LINK Support. Bit 2 - x4 TX SUB-LINK Support. Bit 3 - x8 TX SUB-LINK Support. Bit 4 - x12 TX SUB-LINK Support. Bit 5 - x16 TX SUB-LINK Support. Bit 6 - x32 TX SUB-LINK Support. All other bits are reserved.
5	RX SUB-LINK Width. When LINK Bandwidth Change field (bit 0 of Symbol 1 of TS1) of the TS1 Ordered Set is Clear, bits 0 through 6 indicate that the transmitting Port supports the corresponding RX SUB-LINK width. When LINK Bandwidth Change field (bit 0 of Symbol 1 of TS1) of the TS1 Ordered Set is Set, bits 0 through 6 indicate that the transmitting Port is requesting a change to the corresponding RX SUB-LINK Width. When multiple RX SUB-LINK widths are requested, the Downstream Port chooses between them and indicates it in TS2. Bit 0 - x1 RX SUB-LINK Support. Always set to 1. Support for x1 RX SUB-LINK is mandatory. Bit 1 - x2 RX SUB-LINK Support. Bit 2 - x4 RX SUB-LINK Support. Bit 3 - x8 RX SUB-LINK Support. Bit 4 - x12 RX SUB-LINK Support. Bit 5 - x16 RX SUB-LINK Support. Bit 6 - x32 RX SUB-LINK Support. All other bits are reserved.

Symbol Number	Description
6	Training Control. Bit 0 – Hot Reset - 0b: De-assert; 1b: Assert. Bit 1 – Disable Link- 0b: De-assert; 1b: Assert. Bit 2 – Loopback - 0b: De-assert; 1b: Assert. Bit 3 – Disable Scrambling - 0b: De-assert; 1b: Assert. Bit 4:7 – Reserved. Multiple bits in the Training Control[3:0] are not permitted to be Set and Receiver behavior is undefined if multiple bits in this range are Set.
7 - 15	D10.2 (4Ah) as TS1 Identifier.

Table 8-10: TS2 Ordered Set.

Symbol Number	Description
0	COM (K28.5) for Symbol alignment.
1	LINK Bandwidth and GEAR support Bit 0 – LINK Bandwidth Change 0b: De-assert 1b: Assert Bit 1 - LINK Bandwidth Change Confirm. 0b: De-assert. When this bit is Clear, bits 2 through 4 indicate that the transmitting port supports the corresponding data rate, and bits 0 through 1 of Symbol 2 of indicate that the transmitting port supports the corresponding RATE series. At least one High-Speed GEAR bit and RATE series bit must be Set. 1b: Assert. For Downstream Port: When this bit is Set, in a TS2 transmitted by a Downstream Port, bits 2 through 4 indicate the data rate selected, and bits 0 through 1 indicate the RATE series selected by the Downstream Port. For Upstream Port: When this is Set, in a TS2 transmitted by an Upstream Port, bits 2 through 4 indicate that the Upstream Port supports the corresponding data rate, and bits 0 through 1 of Symbol 2 indicate that the Upstream Port supports the corresponding RATE series. At least one High-Speed GEAR bit and RATE series bit must be Set. Bit 2 - High-Speed GEAR 1. Bit 3 - High-Speed GEAR 2. Bit 4 - High-Speed GEAR 3. All other bits are Reserved.
2	RATE Series Support Bit 0 - RATE-series A. Bit 1 = RATE-series B. All other bits are reserved.
3	TX-LANE Number within LINK for Transmitter and RX-LANE Number within LINK for Receiver.

Symbol Number	Description
4	TX SUB-LINK Width Support. For Downstream Ports: When LINK Bandwidth Change Confirm field (bit 1 of Symbol 1 of TS2) of the TS2 Ordered Set is Clear, bits 0 through 6 indicate the supported TX SUB-LINK widths, when the corresponding bit is Set it indicates that the SUB-LINK width represented by that bit is supported. When LINK Bandwidth Change Confirm field (bit 1 of Symbol 1 of TS2) of the TS2 Ordered Set is Set, for TS2 Ordered Sets transmitted by Downstream Ports bits 0 through 6 indicate the selected TX SUB-LINK widths, when the corresponding bit is Set it indicates that the SUB-LINK width represented by that bit is selected for operation. Only one TX SUB-LINK Width must be selected. For Upstream Ports: Bits 0 through 6 always indicate the supported TX SUB-LINK widths. Bit 0 - x1 TX SUB-LINK Support. Bit 1 - x2 TX SUB-LINK Support. Bit 2 – x4 TX SUB-LINK Support. Bit 3 - x8 TX SUB-LINK Support. Bit 4 - x12 TX SUB-LINK Support. Bit 5 - x16 TX SUB-LINK Support. Bit 6 - x32 TX SUB-LINK Support. All other bits are reserved.

Symbol Number	Description
5	RX SUB-LINK Width Support. For Downstream Ports: When LINK Bandwidth Change Confirm field (bit 1 of Symbol 1 of TS2) of the TS2 Ordered Set is Clear, bits 0 through 6 indicate the supported RX SUB-LINK widths, when the corresponding bit is Set it indicates that the SUB-LINK width represented by that bit is supported. When LINK Bandwidth Change Confirm field (bit 1 of Symbol 1 of TS2) of the TS2 Ordered Set is Set, for TS2 Ordered Sets transmitted by Downstream Ports bits 0 through 6 indicate the selected RX SUB-LINK widths, when the corresponding bit is set it indicates that the SUB-LINK width represented by that bit is selected for operation. Only one RX SUB-LINK Width must be selected. For Upstream Ports: Bits 0 through 6 always indicate the supported RX SUB-LINK widths. Bit 0 - x1 RX SUB-LINK Support. Bit 1 - x2 RX SUB-LINK Support. Bit 2 – x4 RX SUB-LINK Support. Bit 3 - x8 RX SUB-LINK Support. Bit 4 - x12 RX SUB-LINK Support. Bit 5 - x16 RX SUB-LINK Support. Bit 6 - x32 RX SUB-LINK Support. All other bits are reserved.
6	Training Control. Bit 0 – Hot Reset - 0b: De-assert; 1b: Assert. Bit 1 – Disable Link - 0b: De-assert; 1b: Assert. Bit 2 – Loopback - 0b: De-assert; 1b: Assert. Bit 3 – Disable Scrambling - 0b: De-assert; 1b: Assert. Bit 4:7 – Reserved. Multiple bits in the Training Control[3:0] are not permitted to be Set and Receiver behavior is undefined if multiple bits in this range are Set.
7 - 15	D5.2 (45h) as TS2 Identifier.

8.4.2. Electrical Idle

MIPI M-PHY has a TAIL-OF-BURST sequence and line mechanisms to signal entry into electrical idle. The EIOS sequence must be transmitted before the TAIL-OF-BURST sequence. Receipt of TAIL-OF-BURST, with or without EIOS, signals entry into Electrical Idle.

A device is permitted in all speeds of operation to infer Electrical Idle instead of detecting Electrical Idle using analog circuitry. Table 8-11 summarizes the conditions to infer Electrical Idle.

Table 8-11: M-PCIe Electrical Idle Inference Conditions

State	Any one of the following Conditions
L0	Absence of a Flow Control Update DLLP in a 128 µs window. Absence of a SKP Ordered Set on any of the configured RX-LANEs in a 128 µs window. Absence of either a Flow Control Update DLLP[126] or a SKP Ordered Set on any of the configured RX-LANEs in a 128 µs window.

The Electrical Idle exit condition must not be determined based on inference of Electrical Idle condition. For area efficiency, an implementation is permitted to choose to implement a common timeout counter per LTSSM and look for the Electrical Idle inference condition within the common timeout window determined by the common counter for each of the LANEs the LTSSM controls instead of having a timeout counter per LANE.

8.4.3. EIEOS for M-PCIe

EIEOS is not used for M-PCIe, instead the M-PHY PREPARE and SYNC substate protocols are used for HIBERN8 exit and bit and symbol lock.

8.4.4. Lane Polarity Inversion

M-PCIe does not support the detection of polarity inversion since M-PHY does have corresponding support. Implementations are permitted to use the M-PHY transmitter polarity inversion capability. The mechanism to enable polarity inversion is outside the scope of this specification. Transmitter polarity inversion must be configured prior to exit from Detect.Quiet.

8.4.5. Fast Training Sequence (FTS)

Fast training sequences are not defined for M-PCIe. MIPI M-PHY has a specific physical layer sequence for exit from STALL state to BURST state that is defined to achieve bit lock, symbol lock and LANE-to-LANE de-skew.

The (C, NC)_REFCLK_RX_HS_G(1,2,3)_SYNC_LENGTH_Capability advertises the time it takes for a remote receiver to achieve bit lock for the corresponding HS-GEAR for common and non-common clock configurations respectively, this capability is read in Configuration.Start. During

[126] A Flow Control Update DLLP is either an UpdateFC as defined in this specification or an MRUpdateFC as defined in the *Multi-Root I/O Virtualization and Sharing Specification* (MR-IOV).

Initialization the transmitter TX_HS_G(1,2,3)_SYNC_LENGTH field must be programmed, in Configuration.Start to a value which is greater than or equal to the remote Receiver's SYNC length capability for the corresponding HS-GEAR and the clock configuration supported.

During Dynamic Link Re-Configuration, the transmitter must use the TX_HS_G(1,2,3)_SYNC_LENGTH value that corresponds to the Configured LINK Gear.

8.4.6. LINK Data RATE

The LINK Initial Discovery and Configuration process determines the operating High Speed Data rate and SUB-LINK widths along with configuring other M-PHY parameters.

8.4.7. LINK Width

M-PCIe supports the same LINK width configurations as supported by PCI Express. Splitting of ports into multiple links is implementation specific and outside the scope of this specification.

8.4.8. LANE-to-LANE De-skew

The Receiver must compensate for the allowable skew between all Lanes within a multi-Lane Link before delivering the data and control to the Data Link Layer. An unambiguous Lane-to-Lane de-skew mechanism is the MARKER0 Symbol received for symbol synchronization or the COM Symbol of Ordered Sets received during training sequence or SKP Ordered Sets, since Ordered Sets are transmitted simultaneously on all Lanes of a configured Link.
Table 8-12 specifies the required LANE-to-LANE De-skew timing parameters.

Table 8-12: LANE-to-LANE De-skew Timing Parameters

Parameter	Description	Value
$T_{L2L_SKEW_HS_TX}$	Permitted skew between any two LANEs measured at the transmitter's pins.	1600 ps
$T_{L2L_SKEW_HS_RX}$	Permitted skew between any two LANEs measured at the receiver's pins.	4000 ps

8.4.9. LINK Training and Status State Machine (LTSSM)

8.4.9.1. Introduction

The LINK training and status state machine is responsible for multi-LANE management of M-PHY LANEs.

Refer to MIPI M-PHY specification for details about the per LANE M-PHY state machine. M-PCIe specification describes the interaction of the LTSSM with per LANE MIPI M-PHY state machine.

The LTSSM states are illustrated in Figure 8-11.

Detect

This is the initial state after power-on reset and can also be entered from Hot-Reset, Disable, L2, Loopback and Recovery. Detection of DIF-N on RX-LANE(0) indicates that remote end is present.

Configuration

The M-PHY MODULE capabilities are discovered and MODULE attributes configured in this state.

Once the M-PHY parameters and Link capabilities are configured, both the Transmitter and Receiver are sending and receiving data at the configured SUB-LINK width, Rate series and data RATE. Also, LANE-to-LANE de-skew must occur, scrambling can be disabled if permitted, and the Disabled state can be entered.

Recovery

In Recovery, both the Transmitter and Receiver are sending and receiving Training Sequences. Recovery allows a Configured LINK to change Link Bandwidth, re-establish bit lock, Symbol lock, and LANE-to-LANE de-skew. Recovery state is also used for re-configuration of LINK bandwidth.

L0

L0 is the normal operational state where data and control packets can be transmitted and received. All power management states are entered from this state.

While in L0, all configured LANES of the transmit SUB-LINK are permitted to enter STALL state. When one LANE enters STALL all configured LANES of the SUB-LINK must enter STALL.

While in L0, all configured LANES of the receive SUB-LINK are required to support entering STALL state.

SUB-LINK must re-establish bit lock, Symbol lock, and LANE-to-LANE de-skew when exiting from STALL state..

Each SUB-LINK can enter and exit STALL independently.

L1

L1 is intended as a power savings state.

L2

Power can be aggressively conserved in L2. Most of the Transmitter and Receiver may be shut off. Main power and clocks are not guaranteed, but Aux power is available.

Loopback

Loopback is intended for test and fault isolation use. Only the entry and exit behavior is specified, all other details are implementation specific. Loopback must operate on a per Lane basis only and the Loopback Receive LANE must be selected and configured using RRAP.

A Loopback master is the component requesting Loopback.

A Loopback slave is the component looping back the data.

The entry mechanism for a Loopback master is device specific.

Disabled

The intent of the Disabled state is to allow a Configured LINK to be disabled until directed.

Hot Reset

Hot-Reset can only be triggered by a Downstream Port. The Downstream Port uses training sequences to propagate Hot-Reset.

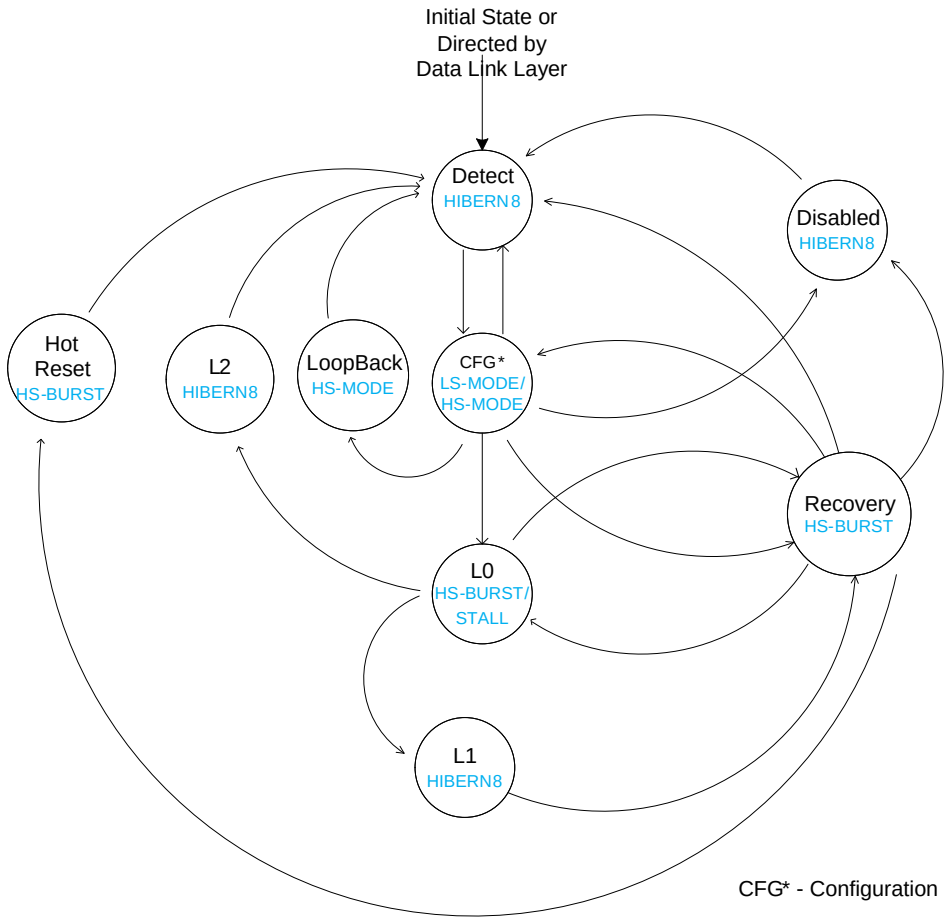


Figure 8-11: M-PCIe LTSSM

Comment [VS359]: Arc from L0 to Recovery; removed bidir arrow.
Mpcie_28August2014 Errata C3

8.4.9.2. Detect

8.4.9.2.1. Detect.Quiet

- If this state is entered through Fundamental Reset assertion, the assertion of Fundamental Reset must cause the Local Reset to M-PHY MODULES to assert, and the corresponding TX-LANEs and RX-LANEs will be in DISABLED state. The de-assertion of Fundamental Reset moves the TX-LANEs and RX-LANEs to HIBERN8 state. It is required that fundamental reset, when applied to the Component on one side of the link should also be applied on the Component on other side of the link to ensure that the two sides are synchronized on reset assertion. If this state is entered from any other LTSSM state due to a timeout error, the M-PHY MODULES must reset to default power-on settings, and the TX-LANEs and RX-LANEs must move to HIBERN8 upon the de-assertion of the Local Reset.
- LinkUp = 0b.
- The idle_to_recovery variable must be set to 0b.
- The recovery_to_configuration variable must be set to 0b.
- LS-MODE must be the mode of operation on exit from this state.
 - TX-LANE(0) and RX-LANE(0) must be configured to operate in LS-MODE, if they are not already in LS-MODE.
 - The next state is Detect.Active when the TX-LANE(0) has resided in HIBERN8 state for at least $T_{HIBERN8}$ since entering this state and one of the following conditions are met:
 - directed or
 - DIF-N is detected on RX-LANE(0) and after DIF-N detection, the TX-LANE(0) is ready to exit HIBERN8.
 - TX-LANE(0) must exit HIBERN8 state on exit from this state.

8.4.9.2.2. Detect.Active

- TX-LANE(0) must be in SLEEP state. The other TX-LANEs and RX-LANEs of the Port must remain in HIBERN8 state.
- The next state is Configuration.Start if the TX-LANE(0) has resided in SLEEP state for at least $T_{ACTIVATE}$ time, where $T_{ACTIVATE}$ is measured after detecting a DIF-N on RX-LANE(0).
- Otherwise, the next state is Detect.Quiet after a 24 ms timeout.
 - RCT must be triggered to move TX-LANE(0) to HIBERN8.

Comment [VS360]: Mpcie_28Aug2014 Errata B3

8.4.9.3. Configuration

The Configuration substate machine is shown in Figure 8-12.

LinkUp state must not be altered in Configuration substates except in Configuration.Idle.

Field Code Changed

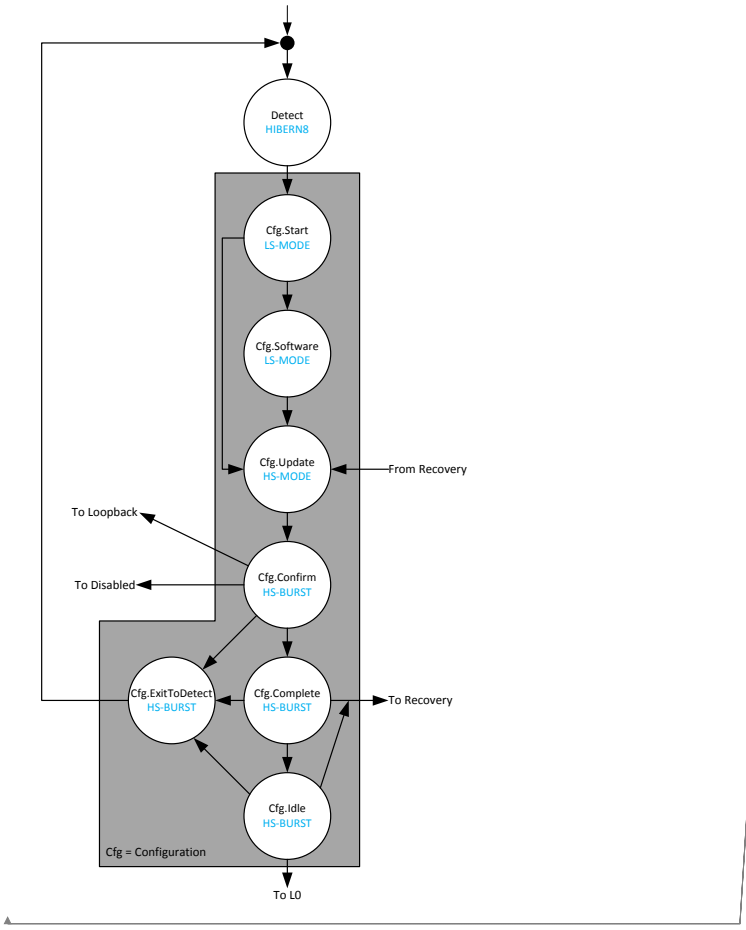


Figure 8-12: Configuration Sub-State Machine

8.4.9.3.1. Configuration.Start

- In this state, TX-LANE(0) and RX-LANE(0) are in SLEEP state. The other TX-LANEs and RX-LANEs of the Port must remain in HIBERN8 state.
- Scrambling must be disabled on entry to this substate.
- The Upstream component must initiate the Initial LINK Discovery and Configuration process.
 - The Upstream component starts a PWM-BURST using TX-LANE(0) and continuously sends COM symbols until it receives at least two consecutive COM symbols on RX-LANE(0). At that point, the Upstream component has completed the COM-COM handshake and the Upstream component is now operating as the RRAP Master (see Section 8.2.3). The Upstream component is permitted to transmit any number of PAD symbols (including 0) between the last COM symbol and the first RRAP Command packet.
 - When the Downstream component detects the start of a PWM-BURST on RX-LANE(0) followed by least two consecutive COM symbols on RX-LANE(0), it must start a PWM-BURST on TX-LANE(0) and continuously send COM symbols until it receives an RRAP Command packet. At that point, the Downstream component has completed the COM-COM handshake, responds to this RRAP Command packet as an RRAP Target and continues to operate the RRAP Target (see Section 8.2.3). The Downstream component is permitted to send any number of PAD symbols (including 0) between the last COM symbol and the first RRAP Response packet.
- The Upstream component performs Local Register Configuration (LRC) and Remote Register Configuration (RRC).
 - Local Register Configuration (LRC) is the implementation-specific protocol used by the Upstream and Downstream components to configure the parameters of their local MODULE. LRC mechanism is outside the scope of the specification.
 - The Upstream component is permitted to enter Configuration.Software after successfully exchanging at least 1 RRAP command and its corresponding response.
 - Remote Register Configuration (RRC) is the process where the Upstream component, uses the RRAP mechanism to configure parameters of the Downstream component's MODULE.
- The Downstream component performs Local Register Configuration (LRC) and responds to RRAP Commands it receives. When LRC is done and it detects the end of PMW-BURST on its RX-LANE(0), the Downstream component terminates the PWM-BURST. Until then, the Downstream component continues the PWM-BURST, transmitting PAD symbols or RRAP Responses as defined by RRAP (see Section 8.2.3).
- The Upstream component continues to operate as an RRAP Master, sending PAD symbols and RRAP Commands as defined by RRAP (see Section 8.2.3) until LRC is done, RRC is done, and the RRAP Response for the last RRAP Command has been received. When all of these conditions have been met, the Upstream component takes the following actions:
 - If the Configuration.Software Supported bit is set in the M-PCIe Capabilities register, it:
 - Continues operating as an RRAP Master

- Continues the PWM-BURST, transmitting PAD symbols
- Scrambling remains disabled
- Next State is Configuration.Software.

o Otherwise the Upstream component:

- Stops operating as an RRAP Master.
- Terminates the PWM-BURST on TX-LANE(0)
- Once both TX-LANE(0) and RX-LANE(0) have entered SLEEP or STALL, RCT must be triggered to move TX-LANE(0) and RX-LANE(0) to HIBERN8. All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.
- Next State is Configuration.Update when all the TX-LANEs and RX-LANEs are in HIBERN8 state.

❑ If the Upstream component is a RRAP Target in the TEST mode, it continues to send PAD symbols and RRAP Responses as defined by RRAP (see Section 8.2.3) until LRC is done and it detects the end of the PWM-BURST on its RX-LANE(0). When both of these conditions have been met, it:

o Stops operating as an RRAP Target.
o Terminates the PWM-BURST on TX-LANE(0)
o Once both TX-LANE(0) and RX-LANE(0) have entered SLEEP or STALL, RCT must be triggered to move TX-LANE(0) and RX-LANE(0) to HIBERN8. All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.
o Next State is Configuration.Update when all the TX-LANEs and RX-LANEs are in HIBERN8 state.

❑ The Downstream component continues to operate as an RRAP Target, sending PAD symbols and RRAP Responses as defined by RRAP (see Section 8.2.3) until LRC is done and it detects the end of the PWM-BURST on its RX-LANE(0). When both of these conditions have been met, it:

o Stops operating as an RRAP Target.
o Terminates the PWM-BURST on TX-LANE(0)
o Once both TX-LANE(0) and RX-LANE(0) have entered SLEEP or STALL, RCT must be triggered to move TX-LANE(0) and RX-LANE(0) to HIBERN8. All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.
o Next State is Configuration.Update when all the TX-LANEs and RX-LANEs are in HIBERN8 state.

8.4.9.3.2. Configuration.Software

This substate is optional and only used by the Upstream component, however please note that some form factor specifications may require this substate to be supported by Upstream component. Support for this substate is indicated by the Configuration.Software Support bit (see Section 7.28.2). Implementations that don't support this state are strongly recommended to provide implementation specific mechanism for system software to access to both local and remote M-PHY registers. When the Upstream component is in this substate, the Downstream component is in the Configuration.Start substate. When the Upstream component exits this substate, both the Upstream and Downstream components enter the Configuration.Update substate.

- In this state, TX-LANE(0) and RX-LANE(0) are in the PWM-BURST state. The other TX-LANEs and RX-LANEs of the Port must remain in HIBERN8 state.
- Scrambling remains disabled on entry to this substate.
- The Upstream component remains in this state as long as the Config bit in the M-PCIe Phy Control Address register is Set (see Section 7.28.6) or an RRAP Command is pending.
- In this substate, a write to the Phy Control Address register with the Read bit Set causes the Upstream component to read the phy register associated with Phy Location, UpperAddr, and LowerAddr.
 - If Phy Location is 1, the Upstream component sends an RRAP Command packet that reads the associated register in the Remote Phy. When the RRAP Response is received, the result is stored in Phy Register Data and the A bit is stored in RRAP Abort. The Phy Control Pending bit is Set when the RRAP Command is sent and is Cleared when either the corresponding RRAP Response packet is received or the RRAP Request times out (see Table 8-5). The Phy Control Error bit is Cleared when the RRAP Command is sent and is Set if an error is detected.
 - If Phy Location is 0, the Upstream component reads the corresponding local phy register. The Phy Register Data field contains the read result. The Phy Control Pending bit is Set when the request is started and Cleared when the request is completed (this may be short so that this may not be visible to software). The Phy Control Error bit is Cleared when the request is started and Set if an error is detected. The RRAP Abort bit is always Cleared.
- In this substate, a write to the Phy Control Data register causes the Upstream component to write the phy register associated with Phy Location, UpperAddr and LowerAddr.
 - If Phy Location is 1, the Upstream component sends an RRAP Command packet that writes the associated register in the Remote Phy with the contents of the Phy Register Data field. The Phy Control Pending bit is Set when the RRAP Command is sent and is Cleared when either the corresponding RRAP Response packet is received or the RRAP Request times out (see Table 8-5). The Phy Control Error bit is Cleared when the RRAP Command is sent and is Set if an error is detected. The A bit in the RRAP Response is stored in the RRAP Abort bit.
 - If Phy Location is 0, the Upstream component writes the corresponding local phy register. The Phy Control Pending bit is Set when the request is started and Cleared when the request is completed (this may be short so that this may not be visible to

software). The Phy Control Error bit is Cleared when the request is started and Set if an error is detected. The RRAP Abort bit is always Cleared.

- Phy Control Error is Set when a read or write operation is requested, Phy Location is 0, and any field in M-PCIe Phy Control Address Register contains a reserved value.
- The Upstream component continues to operate as an RRAP Master, sending PAD symbols and RRAP Commands as defined by RRAP (see Section 8.2.3) until both Phy Control Pending and Config are Clear. When this occurs, the Upstream component:
 - Stops operating as an RRAP Master.
 - Terminates the PWM-BURST on TX-LANE(0)
 - Once both TX-LANE(0) and RX-LANE(0) have entered SLEEP or STALL, RCT must be triggered to move TX-LANE(0) and RX-LANE(0) to HIBERN8. All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.
 - Next State is Configuration.Update when all the TX-LANEs and RX-LANEs are in HIBERN8 state.

8.4.9.3.3. Configuration.Update

- On entry, all the TX-LANEs and RX-LANEs are in HIBERN8 state.
- ~~The recovery_to_configuration variable must be set to 0b.~~
- If the recovery_to_configuration variable is set when entering this state:
 - The recovery_to_configuration variable must be set to 0b
 - Configuration attributes must be set to the values negotiated during reconfiguration
- The Configured TX-LANEs must exit to STALL after residing in this state for $T_{HIBERN8}$.
- The next state is Configuration.Confirm if all Configured TX-LANEs have resided in STALL for at least $T_{ACTIVATE}$, where $T_{ACTIVATE}$ is measured after detecting DIF-N on all the RX-LANEs that are part of the new LINK width.

Comment [VS361]: Mpcie_28August2014 Errata B9

8.4.9.3.4. Configuration.Confirm

- For a Downstream Port, the next state is Disabled if directed by a higher Layer to assert the Disable Link bit (TS1) on all Configured TX-LANEs.
- Next state is LoopbackMaster.Entry if directed, and the Transmitter is capable of being a loopback master, which is determined by implementation specific means.
 - "If directed" applies to a Port that is instructed by a higher Layer to assert the Loopback bit in TS1 on all Configured TX-LANEs.
- The transmitter must transmit TS1 Ordered Sets on all Configured TX-LANEs.
- For an Upstream Port, the next state is Disabled after any Configured RX-LANEs has received TS1 Ordered Sets with the Disable Link bit asserted in two Consecutive TS1 Ordered Sets.
- Next state is Loopback (as loopback slave) after any Configured RX-LANEs has received TS1 Ordered Sets with the Loopback bit asserted in two Consecutive TS1 Ordered Sets.

- The next state is Configuration.Complete after eight Consecutive TS1 or TS2 Ordered Sets are received on all Configured RX-LANEs.
- Otherwise, the next state is Configuration.ExitToDetect after a 2 ms timeout.

8.4.9.3.5. Configuration.Complete

- The transmitter must transmit TS2 Ordered Sets on all Configured TX-LANEs.
- Scrambling must be disabled if, on all Configured RX-LANEs, the Disable Scrambling bit is asserted in two Consecutively received TS2 Ordered Sets, regardless whether the LINK width is symmetric or not.
 - The Port that is sending the Disable Scrambling bit on all of the Configured TX-LANEs must also disable scrambling.
 - Scrambling must remain disabled until the LTSSM transitions through Detect state.
- The next state is Configuration.Idle when all of the following conditions are true:
 - LANE-to-LANE de-skew has been established
 - All Configured RX-LANEs receive eight Consecutive TS2 Ordered Sets, and
 - 16 TS2 Ordered Sets have been transmitted after receiving one TS2 Ordered Set.
- After a 2ms timeout, the next state is determined as follows:
 - The next state is Recovery.Entry if idle_to_recovery variable is 0b.
 - The idle_to_recovery variable must be set to 1b before entering Recovery.Entry.
 - The next state is Configuration.ExitToDetect if idle_to_recovery variable is 1b.

8.4.9.3.6. Configuration.Idle

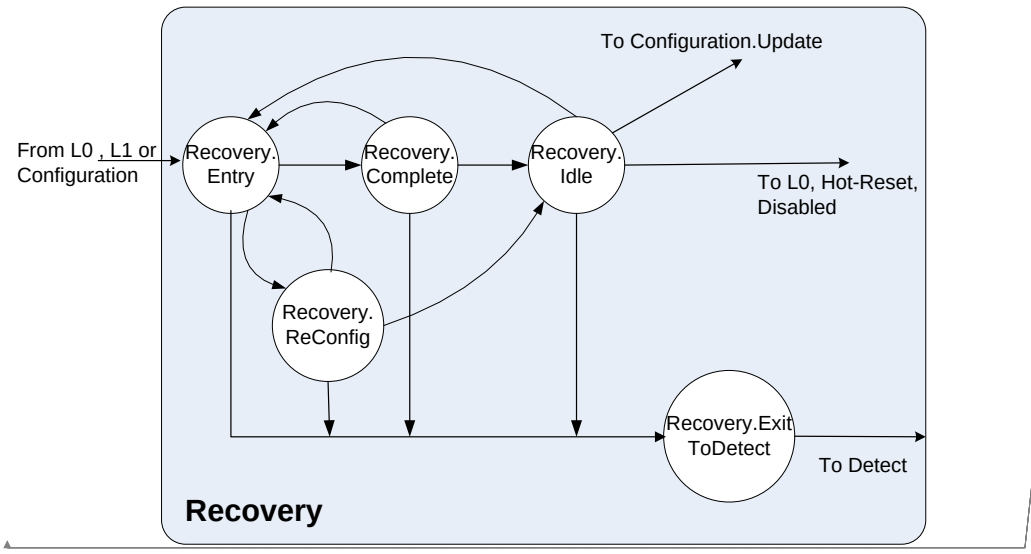
- The Transmitter must start to transmit Idle data Symbols on all Configured TX-LANEs.
- LinkUp must be set to 1b
- The next state is L0 when both of the following conditions are true:
 - Eight Consecutive Symbol Times of Idle data are received on all Configured RX-LANEs, and
 - 16 Idle data Symbols have been transmitted on each LANE after receiving one Idle data Symbol.
- After a 2ms timeout, the next state is determined as follows
 - The next state is Recovery.Entry if idle_to_recovery variable is 0b.
 - The idle_to_recovery variable must be set to 1b before entering Recovery.Entry.
 - The next state is Configuration.ExitToDetect if idle_to_recovery variable is 1b.

8.4.9.3.7. Configuration.ExitToDetect

- The following steps must be followed in sequence:
 - All the Configured TX-LANEs must send at least one EIOS, terminate the HS-BURST and enter STALL.
 - All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10
 - Local Reset must be asserted to all M-PHY MODULES associated with the LINK.
 - The next state is Detect

8.4.9.4. Recovery

The Recovery substate machine is shown in Figure 8-13.



Field Code Changed

Figure 8-13: Recovery Sub-State Machine.

8.4.9.4.1. Recovery.Entry

- The transmitter must transmit TS1 Ordered Sets on all Configured TX-LANEs.
 - If bandwidth change is not requested then transmitter must transmit TS1 Ordered Sets with the LINK Bandwidth Change bit Clear.
 - A component initiating a bandwidth change must transmit TS1 Ordered Sets with the LINK Bandwidth Change bit Set and the requested LINK bandwidth parameters on all Configured TX-LANEs.
 - A component that is not initiating bandwidth change must transmit TS1 Ordered Sets with the LINK Bandwidth Change bit Set upon receiving 2 or more TS1 or TS2 Ordered Sets with the Link Bandwidth Change bit Set in this state . The LINK bandwidth parameters must reflect the supported configurations.
- Next State is Recovery.ReConfig if eight Consecutive TS1 or TS2 Ordered Sets are received on all Configured RX-LANEs with the LINK Bandwidth Change bit set, matching the LINK Bandwidth Change bit on the transmit side.
 - Lane to Lane De-skew must be completed through implementation-specific means before entering Recovery.ReConfig state.
- Next State is Recovery.Complete if eight Consecutive TS1 or TS2 Ordered Sets are received on all Configured RX-LANEs with the LINK Bandwidth Change bit Clear, matching the LINK Bandwidth Change bit on the transmit side.

- Lane to Lane De-skew must be completed through implementation-specific means before entering Recovery. Complete state.

- Otherwise, the next state is Recovery.ExitToDetect after a 24 ms timeout.

8.4.9.4.2. Recovery.ReConfig

For the Downstream Port, the LTSSM next state and the values to use in the TS2 Ordered Set are determined by the Dynamic Link Re-Configuration rules as described in Section 8.7.3. The following rules apply to the Downstream Port.

- If the Downstream Port determines that the LINK Bandwidth must change from current LINK Bandwidth as per the Dynamic LINK Re-configuration rules (Section 8.7.3) then
 - The transmitter transmits TS2 Ordered Sets on all Configured TX-LANEs with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Set.
 - The LINK Bandwidth parameters in the transmitted TS2 Ordered Sets must reflect the new target LINK Bandwidth, determined based on the rules described in Section 8.7.3.
 - The next state is Recovery.Idle after all the Configured RX-LANEs receive eight Consecutive TS2 Ordered Sets with LINK Bandwidth Change bit Set, LINK Bandwidth Change Confirm bit Set, 16 TS2 Ordered Sets are sent with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Set after receiving one TS2 Ordered Set.
 - The recovery_to_configuration variable must be set to 1b.
- If the Downstream Port determines that the LINK Bandwidth must not change from current LINK Bandwidth as per the Dynamic LINK Re-configuration rules (Section 8.7.3) then
 - The transmitter must transmit TS2 Ordered Sets on all Configured TX-LANEs with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Clear.
 - The LINK Bandwidth parameters in the transmitted TS2 Ordered Sets must match the values used in TS1 Ordered Set.
 - The next state is Recovery.Idle after all the Configured RX-LANEs receive eight Consecutive TS2 Ordered Sets with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Clear, and 16 TS2 Ordered Sets are transmitted with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Clear after receiving one TS2 Ordered Set.
- After a 2 ms timeout, the next state is determined as follows
 - The next state is Recovery.Entry if idle_to_recovery variable is 0b.
 - The idle_to_recovery variable must be set to 1b before entering Recovery.Entry.
 - The next state is Recovery.ExitToDetect if idle_to_recovery variable is 1b.

For the Upstream port, the LTSSM next state and the values to use in the TS2 Ordered Set to be transmitted are determined by the values in received TS2 Ordered Set. The following rules apply to the Upstream Port.

- The LINK Bandwidth parameters in the transmitted TS2 Ordered Sets must match the values used in TS1 Ordered Sets issued by the Upstream Port.
- The transmitter must transmit TS2 Ordered Sets on all Configured TX-LANEs with the LINK Bandwidth Change bit Set, and LINK Bandwidth Change Confirm bit Clear, until it receives 2 Consecutive TS2 Ordered Sets with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Set on all Configured RX-LANEs.
 - Once the above condition is met the transmitter must start to transmit TS2 Ordered Sets on all Configured TX-LANEs with the LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Set.
 - The Upstream Port must accept the new target LINK Bandwidth if two Consecutive TS2 Ordered Sets were received with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Set.
- The next state is Recovery.Idle after all the Configured RX-LANEs receive eight Consecutive TS2 Ordered Sets with LINK Bandwidth Change bit Set, LINK Bandwidth Change Confirm bit Set, 16 TS2 Ordered Sets are sent with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Set after receiving one TS2 Ordered Set.
 - The recovery_to_configuration variable must be set to 1b.
- Next state is Recovery.Idle after all the Configured RX-LANEs receive eight Consecutive TS2 Ordered Sets with the LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Clear , and 16 TS2 Ordered Sets are sent with LINK Bandwidth Change bit Set and LINK Bandwidth Change Confirm bit Clear after receiving one TS2 Ordered Set.
- After a 2ms timeout, the next state is determined as follows
 - The next state is Recovery.Entry if idle_to_recovery variable is 0b.
 - The idle_to_recovery variable must be set to 1b before entering Recovery.Entry.
 - The next state is Recovery.ExitToDetect if idle_to_recovery variable is 1b.

8.4.9.4.3. Recovery.Complete

- The transmitter must transmit TS2 Ordered Sets on all Configured TX-LANEs.
- The next state is Recovery.Idle when all the Configured RX-LANEs satisfy both of the following conditions:
 - Eight Consecutive TS2 Ordered Sets are received, and
 - 16 TS2 Ordered Sets have been transmitted after TS2 Ordered Set was received.
- After a 2ms timeout, the next state is determined as follows
 - The next state is Recovery.Entry if idle_to_recovery variable is 0b.
 - The idle_to_recovery variable must be set to 1b before entering Recovery.Entry.
 - The next state is Recovery.ExitToDetect if idle_to_recovery variable is 1b.

8.4.9.4.4. Recovery.Idle

- For a Downstream Port, the next state is Disabled if directed by a higher Layer to assert the Disable Link bit (TS1 and TS2) on all Configured TX-LANEs.
- For a Downstream Port, the next state is Hot Reset if directed by a higher Layer to assert the Hot Reset bit (TS1) all Configured TX-LANEs.
- For Upstream Ports, the Next state is Disabled after any Configured RX-LANEs has Consecutively received two TS1 Ordered Sets with the Disable Link bit asserted.
- For Upstream Ports, the Next state is Hot Reset after any Configured RX-LANEs has received two Consecutive TS1 Ordered Sets with the Hot Reset bit asserted.
- Unless Ordered Sets are being transmitted, Idle data must be transmitted on all Configured TX-LANEs.
 - It is permitted that no idle data be transmitted before transitioning out of this state to Disabled or Hot Reset.
- Next state is L0 if the following conditions are satisfied
 - Eight Consecutive Symbol Times of Idle data are received on all Configured Lanes and 16 Idle data Symbols are sent after receiving one Idle data Symbol.
 - The recovery_to_configuration variable is 0b.
- Next state is Configuration.Update if all of the following conditions are satisfied
 - The recovery_to_configuration variable is 1b.
 - Eight Consecutive Symbol Times of Idle data are received on all Configured Lanes and 16 Idle data Symbols are sent after receiving one Idle data Symbol.
 - All the Configured TX-LANEs and RX-LANEs have entered HIBERN8 state after the above condition is satisfied.
 - All the Configured TX-LANEs must enter STALL state.
 - All Configured LANEs must initiate the process to enter HIBERN8 as described in Section 8.4.10.
- After a 2ms timeout, the next state is determined as follows
 - The next state is Recovery.Entry if idle_to_recovery variable is 0b.
 - The idle_to_recovery variable must be set to 1b before entering Recovery.Entry.
 - The next state is Recovery.ExitToDetect if idle_to_recovery variable is 1b.

8.4.9.4.5. Recovery.ExitToDetect

- The following steps must be followed in sequence:
 - All the Configured TX-LANEs must send at least one EIOS, terminate the HS-BURST and enter STALL.
 - All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10
- Local Reset must be toggled to transition all M-PHY MODULES associated with the LINK through DISABLED state to HIBERN8 state.
 - The next state is Detect

8.4.9.5. L0

- LinkUp must be set to 1b.
 - On receipt of an STP or SDP Symbol, the idle_to_recovery variable is reset to 0b
- Next state is Recovery, with the Transmitter permitted, but not required, to complete any TLP or DLLP in progress, if any of the following conditions are true:
 - Directed to change the LINK Bandwidth for implementation-specific reasons, and the LINK is in DL_Active state
 - The Retrain Link bit in the Link Control register has been Set, or directed to this state by a higher layer for implementation specific reasons.
 - A TS1 or TS2 Ordered Set is received on any Configured RX-LANE.
 - If Electrical Idle is inferred as described in Section 8.4.2 on all LANES.
- Next state is L1 if both Ports of the LINK have completed the L1 entry protocol, for either PCI-PM (see Section 5.3.2.1) or ASPM-L1 (see Section 5.4.1.2.1) and have agreed to enter L1 immediately after the condition for both the receipt and transmission of the EIOS is met.
 i. An EIOS is received on any RX-LANE and
 ii. An EIOS is transmitted on all the Configured TX-LANEs
 - One side of the LINK always initiates entry into L1 by transmitting the EIOS, followed by a transition to STALL state on all configured TX-LANEs. The same Port then waits for the receipt of EIOS on any LANE and then immediately transitions to L1. Conversely, the side of the LINK that first receives the EIOS on any LANE must send an EIOS, followed by a transition to STALL state on all configured TX-LANEs and immediately transition to L1
- Next state is L2 if both Ports on the LINK have completed the L2 entry protocol (see Section 5.3.2.3) and have agreed to enter L2 immediately after the condition for both the receipt and transmission of the EIOS is met.
 i. An EIOS is received on any RX-LANE and
 ii. An EIOS is transmitted on all the Configured TX-LANEs

- One side of the LINK always initiates entry into L2 by transmitting the EIOS, followed by a transition to STALL state on all configured TX-LANEs. The same Port then waits for the receipt of EIOS on any LANE and then immediately transitions to L2. Conversely, the side of the LINK that first receives the EIOS on any LANE must send an EIOS, followed by a transition to STALL state on all configured TX-LANEs and immediately transition to L2.

- The Transmitter is permitted to move all Configured TX-LANES to STALL state while in L0, based on implementation-specific criteria
 - The Transmitter must send at least one EIOS on all Configured TX-LANEs, then terminate the HS-BURST on all Configured TX-LANEs.
- The Transmitter is permitted to move all Configured TX-LANES from STALL state to HS-BURST while in L0, based on implementation-specific criteria
 - Upon entering HS-BURST, the transmitter must send 4 consecutive COM symbols, followed by a single SKP Ordered Set on all Configured TX-LANEs.

8.4.9.6. L1

8.4.9.6.1. L1.Entry

- All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10
- Next state is L1.Idle once all Configured TX-LANEs and RX-LANEs enter HIBERN8 state.

8.4.9.6.2. L1.Idle

- All the Configured TX-LANEs and RX-LANEs must remain in the HIBERN8 state.
- It is permitted for any PLL(s) to be shut down and the M-PHY to be power gated in this state.
- The next state is L1.Exit when all the Configured TX-LANEs have resided in this state for at least $T_{HIBERN8}$ time since entering this state and one of the following conditions are met:
 - A specified or implementation-specific condition requires the L0 state, or
 - A DIF-N state is detected on any Configured RX-LANE.

8.4.9.6.3. L1.Exit

- All Configured TX-LANEs must be placed into the STALL state.
- The next state is Recovery after all Configured TX-LANEs have resided in STALL state for at least $T_{ACTIVATE}$, where $T_{ACTIVATE}$ is measured after detecting DIF-N on all the Configured RX-LANEs.

8.4.9.7. L2

8.4.9.7.1. L2.Entry

- All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10
- Next state is L2.Idle once all Configured TX-LANEs and RX-LANEs enter HIBERN8 state.

8.4.9.7.2. L2.Idle

- All the Configured TX-LANEs and RX-LANEs must remain in the HIBERN8 state.
- For Downstream Port:
 - The next state is Detect when directed.
 - Main power must be restored before being directed to Detect.
 - All the Configured TX-LANEs must reside in HIBERN8 state for at least $T_{HIBERN8}$ prior to being directed out of L2 state.
- For Upstream Port:
 - The next state is Detect when DIF-N is detected on any Configured RX-LANEs.
 - The Configured TX-LANEs must remain in HIBERN8 state.

8.4.9.8. Disabled

- For both the Upstream and Downstream Ports
 - All Configured TX-LANEs transmit 16-32 TS1 Ordered Sets with the Disable Link bit asserted, followed by at least two consecutive EIOS and then transition to STALL state.
 - All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10
 - The LINK is considered to be Disabled when all LANEs enter HIBERN8.
 - LinkUp = 0b.
- For upstream port, the next state is Detect when DIF-N is detected on any Configured RX-LANEs.
 - All the Configured TX-LANEs must remain in HIBERN8 state.
- For downstream port, the next state is Detect when directed.
 - All the Configured TX-LANEs and RX-LANEs must remain in HIBERN8.

8.4.9.9. Loopback

The Loopback state is further subdivided into LoopbackMaster and LoopbackSlave states as defined below. Section 8.4.9.9.1 to Section 8.4.9.9.3 covers the LTSSM state transitions for LoopbackMaster. Section 8.4.9.9.4 to Section 8.4.9.9.6 covers the LTSSM state transitions for LoopbackSlave.

8.4.9.9.1. LoopbackMaster.Entry

- LinkUp = 0b
- Loopback master requirements:
 - Transmit TS1 Ordered Sets with the Loopback bit asserted.
 - If all the Configured Loopback LANEs receive two Consecutive TS1 Ordered Sets with Loopback bit asserted
 - The Loopback master transitions all its TX LANES to STALL after it's RX LANEs have transitioned to STALL
 - After a delay of LINK_MIN_SAVE_CONFIG_TIME, the Loopback master transitions its TX LANEs to BURST mode and transmits COM symbols.
 - Next state is LoopbackMaster.Active if a SKP OS has been transmitted on all the Configured Loopback LANEs after receiving at least one COM symbol.
 - Next state is LoopbackMaster.Exit after an implementation specific timeout of less than 100 ms.

8.4.9.9.2. LoopbackMaster.Active

- The Loopback master must send valid 8b/10b encoded Symbols. The Loopback master must not send EIOS except to initiate Loopback exit.
- The next state of the Loopback master is LoopbackMaster.Exit if directed.
- In a symmetric LINK, TX-LANE(M) is used to transmit loopback data received on RX-LANE(M), where M is the LANE number.
- In an asymmetric link, loopback data shall be returned from a given RX-LANE using an optional implementation specific scheme.
 - ~~Using the LOOPBACK_PARTNER_LANE and M-PHY_ANALOG_LOOPBACK_MODE_LANES Test Registers where the number of RX-LANEs exceeds the number of TX-LANEs.~~
 - ~~Implementation specific in the remote partner where the number of TX-LANEs exceeds the number of RX-LANEs.~~

Comment [VS362]: Mcpie_28August2014 Errata A2

Comment [VS363]: A2

8.4.9.9.3. LoopbackMaster.Exit

- The Loopback master must transmit eight consecutive EIOS and all the configured Loopback master TX-LANEs must enter STALL state.
 - The Loopback master TX-LANE must remain in STALL state for 2 ms.
 - Note: The EIOS can be useful in signifying the end of transmit and compare operations that occurred by the Loopback master. Any data received by the Loopback master after any EIOS is received must be ignored since it is undefined.
- The next state of the Loopback master is Detect.
 - All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10

8.4.9.9.4. LoopbackSlave.Entry

- LinkUp = 0b
- Loopback Slave requirements:
 - Transmit TS1 Ordered Sets with Loopback bit set.
 - On transmitting at least 16 TS1 with Loopback bit set, after receiving the first TS1 with Loopback bit set, Loopback slave transitions all its Configured TX LANEs to STALL. Before transitioning TX LANEs to STALL, the Loopback Slave must have prepared all its configured TX and RX LANEs for loopback. Additional TS1s are permitted to be transmitted during local M-PHY configuration.
 - Next state is LoopbackSlave.Active when all the Configured RX-LANEs receive at least one COM symbol.
 - The slave's transmitter must transition to transmitting looped-back data on a Symbol boundary, but it is permitted to truncate any Ordered Set in progress.
 - In a symmetric LINK, TX-LANE(M) is used to transmit loopback data received on RX-LANE(M), where M is the LANE number.
 - The TX-LANE(s) used for transmitting loopback data from a given RX-LANE in an asymmetric LINK shall be determined using an optional implementation specific scheme. ~~as follows:~~
 - ~~Using the LOOPBACK_PARTNER_LANE and M-PHY_ANALOG_LOOPBACK_MODE_LANES Test Registers where the number of RX-LANEs exceeds the number of TX-LANEs.~~
 - ~~Implementation specific where the number of TX-LANEs exceeds the number of RX-LANEs.~~

Comment [VS364]: Mcpie_28August2014 Errata A2

8.4.9.9.5. LoopbackSlave.Active

- A Loopback slave is required to retransmit the received encoded information as received, while continuing to perform clock tolerance compensation:
 - SKPs must be added or deleted on a per Lane basis with the exception that SKPs do not have to be simultaneously added or removed across a Configured SUB-LINK.
 - If a SKP Ordered Set retransmission requires adding a SKP Symbol to accommodate timing tolerance correction, the SKP Symbol is inserted in the retransmitted Symbol stream anywhere adjacent to a SKP Symbol in the SKP Ordered Set following the COM Symbol. The inserted SKP Symbol must be of the same disparity as the received SKPs Symbol(s) in the SKP Ordered Set.
 - If a SKP Ordered Set retransmission requires dropping a SKP Symbol to accommodate timing tolerance correction, the SKP Symbol is simply not retransmitted.

- No modifications of the received encoded symbol are allowed by the Loopback slave even if it is determined to be an invalid encoding (i.e., no legal translation to a control or data value possible).

- Next state of the Loopback slave is LoopbackSlave.Exit when any configured RX-LANE enters STALL state.

8.4.9.9.6. LoopbackSlave.Exit

- All of the configured Loopback slave TX-LANEs must enter STALL state after all Symbols that were received prior to the Loopback slave RX-LANE entering STALL are transmitted.
 - The Loopback slave TX-LANE must remain in STALL state for 2 ms.
- The next state of the Loopback slave is Detect.
 - All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.

8.4.9.10. Hot Reset

- For the Port that is directed by a higher layer to initiate Hot Reset:
 - All the Configured TX-LANEs transmit TS1 Ordered Sets with Hot Reset bit asserted.
 - If two Consecutive TS1 Ordered Sets are received on any Configured RX-LANEs with Hot Reset bit asserted, then:
 - LinkUp = 0b.
 - If the higher layer is no longer directing Hot Reset, the next state is Detect once all the TX-LANEs and RX-LANEs have transitioned to HIBERN8.
 - All the Configured TX-LANEs must send at least one EIOS, terminate the HS-BURST and enter STALL.
 - All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.
 - Otherwise, all the Configured TX Lanes continue to transmit TS1 Ordered Sets with the Hot Reset bit asserted.
 - Otherwise, the next state is Detect after a 2 ms timeout and all the Configured TX-LANEs and RX-LANEs have transitioned to HIBERN8.
 - All the Configured TX-LANEs must send at least one EIOS, terminate the HS-BURST and enter STALL.
 - All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.
- For the Port that is receiving Hot Reset:
 - LinkUp = 0b.
 - All the Configured TX-LANEs transmit TS1 Ordered Sets with the Hot Reset bit asserted.

- If two Consecutive TS1 Ordered Sets were received with the Hot Reset bit asserted, the state continues to be Hot Reset and the 2 ms timer is reset.
- Otherwise, the next state is Detect after a 2 ms timeout and all the TX-LANEs and RX-LANEs have transitioned to HIBERN8.
 - All the Configured TX-LANEs must send at least one EIOS, terminate the HS-BURST and enter STALL.
- All Configured LANEs initiate the process to enter HIBERN8 as described in Section 8.4.10.

8.4.10. Entry to HIBERN8

The following steps must be taken to put all Configured LANEs in HIBERN8.

- All Configured TX-LANEs must initiate entry to SLEEP/STALL.
- Once all Configured RX-LANEs have entered SLEEP/STALL.
 - All Configured RX-LANEs must trigger RCT to enter HIBERN8 state no later than TRX_RCT_to_HIBERN8.
- Once all Configured TX-LANEs and RX-LANEs have entered SLEEP/STALL.
 - All Configured TX-LANEs must trigger RCT to enter HIBERN8 state no earlier than TTX_RCT_to_HIBERN8.
- The following table specifies the timing parameters required for RCT when entering HIBERN8 state.

Table 8-13: HIBERN8 Timing Parameters

Parameter	Value
$T_{RX_RCT_to_HIBERN8}$	800 ns
$T_{TX_RCT_to_HIBERN8}$	1 us

8.5. Receiver Error

A Receiver is permitted to optionally report an 8b/10b Error in the M-PCIe Lane Error Status Register, when allowed to report the error as a Receiver Error as shown in Table 8-14.

IMPLEMENTATION NOTE

Receiver Errors During Configuration and Recovery States

Allowing Receiver Errors to be set while in Configuration or Recovery is intended to allow implementations to report Link Errors that occur while processing packets in those states. For

Comment [VS365]: Mpcie_28Aug2014 Errata A1

Comment [VS366]: Errata A1

Comment [VS367]: Errata A1

Formatted: Font color: Black

example, if the LTSSM transitions from L0 to Recovery while a TLP is being received, a Link Error that occurs after the LTSSM transition can be reported.

Table 8-14: Receiver Error

LTSSM State	Receiver Error
Detect	No action
Configuration	Set on 8b/10b Error.
Recovery	Optionally set on Link Error.
L0	Set on Link Error.
L1	No action
L2	No action
Disabled	Optional: Set on 8b/10b Error
Loopback	No action
Hot Reset	Optional: Set on 8b/10b Error

8.6. Clock Tolerance Compensation

The requirements for clock tolerance compensation are stated in Section 4.2.7 for 8b/10b encoding.

If 2K_PPM_DISABLED (Section 8.2.2) is Set, then SKP Ordered Set must be scheduled for transmission at an interval between 1180 and 1538 ~~symbols~~ Symbol Times, otherwise SKP Ordered Set must be scheduled for transmission at an interval of less than or equal to ~~after every~~ 250~~symbols~~ Symbol Times.

The SRIS Capability must not be implemented when the ~~Link~~ Port is ~~operating with~~using an M-PHY based Physical Layer.

Comment [VS368]: Mpcie_28August2014 Errata B15

Comment [VS369]: B15

Comment [VS370]: B15

Comment [VS371]: Mpcie_28August2014 Errata C8

8.7. Dynamic LINK Bandwidth Management

Dynamic LINK Re-Configuration is an optional normative functionality that supports changing the LINK Bandwidth by changing the RATE series or High-Speed Gears identically on MODULES on both sides of the LINK, and/or changing the number of Configured LANEs per SUB-LINK. A LINK Width change is permitted to be asymmetric (i.e. a MODULE is permitted to be configured with different number of TX-LANEs and RX-LANEs).

Software is notified of any RATE series or LINK speed and LINK width changes through the Link Bandwidth Notification Mechanism as described in PCI Express Specification.
Functions that implement Link Bandwidth Notification Capability must report all bandwidth changes as Link Autonomous Bandwidth change. The Link Bandwidth Management Status bit in the Link Status register must be set when Link retraining has completed following a write of 1b to the Retrain Link bit.

8.7.1. LINK Rate Series and Speed Management

This Section describes how LINK RATE series and LINK speed management is coordinated between the M-PCIe LTSSM, and the software Link observation and control mechanisms as described in PCI Express Base Specification.

If the Hardware_Autonomous_RATE series Re-Configuration Enable bit in the M-PCIe Configuration Attributes is Set, the component is permitted to autonomously adjust the LINK RATE series and use the Dynamic LINK Re-Configuration functionality to request RATE series change.
If the Hardware Autonomous Speed Disable bit in the Link Control 2 register is Clear, the component is permitted to autonomously adjust the LINK High-Speed GEARs using implementation specific criteria.
Either component is permitted to lower or increase the LINK speed by removing or adding the corresponding LINK High-Speed GEAR from the list of supported High-Speed GEARs advertised in the training sets the component transmits. The criteria for removing or adding High-Speed GEARs are implementation specific.
Software is permitted to restrict the maximum speed of LINK operation and set the preferred LINK speed by setting the value in the M-PCIe Target Link Speed Control field and the Target Link Speed field in the Upstream component. After modifying the value in the Target Link Speed field, software must trigger Link retraining by writing 1b to the Retrain Link bit.

8.7.2. LINK Width Management

If the Hardware Autonomous Width Disable bit in the Link Control register is Clear, the component is permitted to autonomously request SUB-LINK width change using implementation specific criteria. Either component is permitted to lower or increase the SUB-LINK width by removing or adding the corresponding TX-LANE width and RX-LANE width setting from the list of supported SUB-LINK width settings advertised in the training sets the component transmits. The criteria for removing or adding LINK width settings are implementation specific.

8.7.3. Dynamic LINK Re-Configuration

A component initiates Dynamic LINK Re-Configuration by directing its LTSSM to the Recovery state and transmitting TS1 Ordered Sets with the LINK Bandwidth Change bit Set, and with the fields in the Training Ordered Set for the requested RATE series, High-Speed GEARs, TX-LANE width and RX-LANE width. Components are not permitted to change the requested bandwidth settings in the training Ordered Sets once Dynamic LINK Re-configuration process is started.

The component receiving the bandwidth change request must respond by transmitting Training Ordered Set TS1 with the LINK Bandwidth Change bit Set. The TS1 fields in this response must be set according to the support for RATE series, High-Speed GEARs and SUB-LINK width.

The Upstream component is responsible for determining the new RATE series, High-Speed GEARs and SUB-LINK width as per the Link Re-configuration Rules as described below . The result of Dynamic LINK Re-Configuration must be communicated by setting the status of the LINK Bandwidth Change and LINK Bandwidth Change Confirm bit in the TS2 Ordered Set transmitted by the Upstream component. The LINK Bandwidth Change Confirm bit when Set

indicates that the LINK must operate at bandwidth different than current bandwidth, and when Clear indicates that the LINK must operate at current bandwidth.

When a component's attempt to negotiate to a particular LINK bandwidth is rejected, that component is not permitted to attempt dynamic LINK re-configuration, until 200 ms has passed from return to L0 following the previous attempt, or until the other component on the Link initiates LINK bandwidth request.

Formatted: Font color: Auto

IMPLEMENTATION NOTE

EXAMPLE OF DOWNSTREAM PORT INITIATED LINK RE-CONFIGURATION

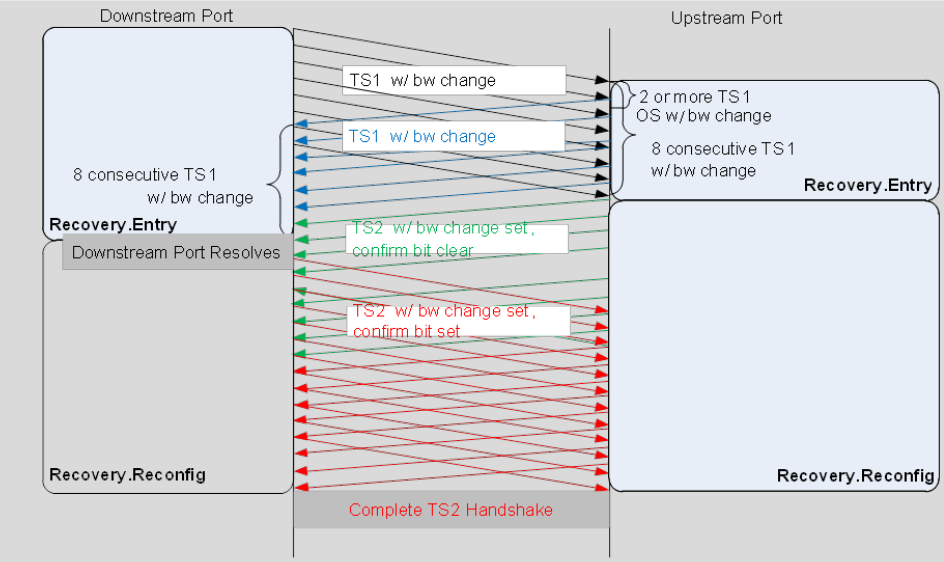


Figure 8-14: Example of Downstream Port initiated LINK Re-Configuration

In the above Example the Downstream Port initiates Link Re-Configuration by transmitting TS1 Ordered Sets with LINK Bandwidth Change bit set. The Upstream Port responds by returning TS1 Ordered Sets with the LINK Bandwidth Change bit Set. After receiving 8 TS1 with LINK Bandwidth Change bit Set, Upstream Port starts transmitting TS2 with the LINK Bandwidth Change bit Set, LINK Bandwidth Change Confirm bit Clear. After receiving 8 TS1 or TS2 Ordered Sets the downstream port determines if bandwidth change is possible based on the parameters received in OS. Meanwhile the Downstream Port continues to drive TS1 Ordered Set with LINK Bandwidth Change bit Set. In this example the Downstream Port determines that bandwidth change is feasible and transitions to Recovery.Reconfig state. Downstream Port begins transmitting TS2 Ordered Sets with both the LINK Bandwidth Change bit and the LINK Bandwidth Change Confirm bit Set. On receipt of TS2 Ordered Set, the upstream port responds by transmitting TS2 Ordered Set with both LINK Bandwidth Change and the LINK Bandwidth Change Confirm bit Set.

As part of the dynamic LINK Re-Configuration, the Upstream component must determine the new target High Speed GEAR, RATE series and Width by comparing the values in the TS1s with the LINK Bandwidth Change bit set received from the Downstream component with the Upstream components supported values, independent of the values the Upstream component may have used in TS1 to request a LINK Bandwidth Change.

- The new High Speed GEAR must be the highest commonly supported High Speed GEAR. If there is no highest commonly supported High Speed Gear then the new High Speed GEAR must remain at the current High Speed GEAR.
- The new SUB-LINK width must be the highest commonly supported SUB-LINK width. If there is no highest commonly supported SUB-LINK width then the new SUB-LINK width must remain at the current SUB-LINK width.
- If the new RATE series requested is different than the current RATE series, and the components on both sides of the LINK support the new RATE series, the new RATE series must be the requested RATE series.
 - When requesting to change the RATE series, the requesting component must set the corresponding RATE series support bit for the new RATE series within the Training Sequences and clear the RATE series support bit for the current RATE series.
 - The responding component must set the corresponding RATE series support bits for all RATE series that it supports.
- If the new RATE series requested is different than the current RATE series, and the component on the other side of the LINK does not support the new RATE series, the new RATE series must remain at the current RATE series.

It is permitted to request High Speed GEAR, RATE series and SUB-LINK width changes at the same time.

It is permissible for the Upstream component to accept a sub-set of the requested bandwidth change parameters. Parameters which are unchanged will be set to the current operating parameters.

Comment [VS372]: Mpcie_28August2014 Errata B9

8.8. M-PHY Registers

8.8.1. M-PHY Capability Registers

The following tables list the set of capabilities that are mandated for M-TX and M-RX MODULEs respectively. Ports are required to support the M-TX and M-RX capabilities required in this section.

Table 8-15: M-TX Capability Attributes

Attribute Name	AttributeID	Description
TX_HSMODE_Capability	0x01	"0b": HS-MODE is not supported. "1b": HS-MODE is supported. Value: 1b.
TX_HSGEAR_Capability	0x02	"01b": Supports HS_G1 "10b": Supports HS_G1 and HS_G2. "11b": Supports HS_G1, HS_G2 and HS_G3. Value: Implementation Specific.
TX_PWMGEAR_Capability	0x04	"001b": Supports PWM_G1. "010b": Supports PWM_G1 to PWM_G2. "011b": Supports PWM_G1 to PWM_G3. "100b": Supports PWM_G1 to PWM_G4. "101b": Supports PWM_G1 to PWM_G5. "110b": Supports PWM_G1 to PWM_G6. "111b": Supports PWM_G1 to PWM_G7. Value: ≥ 001b.
TX_Amplitude_Capability	0x05	"01b": SMALL AMPLITUDE ONLY. "10b": LARGE AMPLITUDE ONLY. "11b": LARGE AND SMALL AMPLITUDE. Value: Implementation Specific.
TX_ExternalSYNC_Capability	0x06	"0b": No support for external SYNC pattern. "1b": Supports external SYNC pattern. Value: Implementation Specific.

<table>
<tr><th>Attribute Name</th><th>AttributeID</th><th>Description</th></tr>
<tr><td>TX_HS_Unterminated_LINE_ Drive_Capability</td><td>0x07</td><td>“0b”: No.
“1b”: Yes.

Value: Implementation Specific.</td></tr>
<tr><td>TX_LS_Terminated_LINE_ Drive_Capability</td><td>0x08</td><td>“0b”: No.
“1b”: Yes.

Value: Implementation Specific.</td></tr>
<tr><td>TX_Min_SLEEP_NoConfig_ Time_Capability</td><td>0x09</td><td>The time the protocol layer must wait in SLEEP state before initiating exit if inline configuration is not done in the previous LS-MODE BURST.

1-15 SI.

<table><tr><th></th><th>3 Mbps</th><th>9 Mbps</th></tr><tr><td>1 SI</td><td>3.33 us</td><td>1.11 us</td></tr><tr><td>15 SI</td><td>50 us</td><td>16.67 us</td></tr></table>
Value: ≤ 15 SI.</td></tr>
<tr><td>TX_Min_STALL_NoConfig_ Time_Capability</td><td>0x0A</td><td>The time the protocol layer must wait in STALL state before initiating exit if inline configuration is not done in the previous HS-BURST.

1-255 SI.

<table><tr><th></th><th>1.25 Gbps</th><th>2.5 Gbps</th><th>5 Gbps</th></tr><tr><td>1 SI</td><td>8 ns</td><td>4 ns</td><td>2 ns</td></tr><tr><td>255 SI</td><td>2040 ns</td><td>1020 ns</td><td>510 ns</td></tr></table>
Value: ≤ 16.</td></tr>
</table>

Attribute Name	AttributeID	Description
TX_Min_SAVE_Config_ Time_Capability	0x0B	The time the protocol layer must wait in SAVE state before initiating exit if inline configuration is done in the previous BURST. 1-250 (in 40 ns steps). 40 ns – 10 us. ~~Value: ≤ 125 (5 us).~~ Value: Implementation Specific
TX_REF_CLOCK_ SHARED_Capability	0x0C	“0b”: No “1b”: Yes Value: Implementation Specific.
TX_PHY_MajorMinor_Release_ Capability	0x0D	[7:4]: Major version number, 0 to 9. [3:0]: Minor version number, 0 to 9. Value: PHY Specific.
TX_PHY_Editorial_Release_ Capability	0x0E	1 to 99. Value: PHY Specific.
TX_Hibern8Time_Capability	0x0F	When entering HIBERN8 from LS-MODE or HS-MODE, the protocol layer must not request exit before a minimum of THIBERN8, which is defined as the larger of local Tx_Hibern8Time_Capability and remote Rx_Hibern8Time_Capability. 1 to 128. Value: ≤ 100 us This attribute is not used by M-PCIe. M-PCIe must use the LINK_HIBERN8_TIME_CAPABILITY attribute as defined in Table 8-2.

Comment [VS373]: Mpcie_28August2014 Errata B6

Table 8-16: M-RX Capability Attributes

Attribute Name	AttributeID	Description
RX_HSMODE_Capability	0x81	"0b": HS-MODE is not supported. "1b": HS-MODE is supported. Value: 1b.
RX_HSGEAR_Capability	0x82	"01b": Supports HS_G1 "10b": Supports HS_G1 and HS_G2. "11b": Supports HS_G1, HS_G2 and HS_G3. Value: Implementation Specific.
RX_PWMGEAR_Capability	0x84	"001b": Supports PWM_G1. "010b": Supports PWM_G1 to PWM_G2. "011b": Supports PWM_G1 to PWM_G3. "100b": Supports PWM_G1 to PWM_G4. "101b": Supports PWM_G1 to PWM_G5. "110b": Supports PWM_G1 to PWM_G6. "111b": Supports PWM_G1 to PWM_G7. Value: ≥ 001b.
RX_HS_Unterminated _Capability	0x85	"0b": No. "1b": Yes. Value: Implementation Specific.
RX_LS_Terminated _Capability	0x86	"0b": No. "1b": Yes. Value: Implementation Specific.
RX_Min_SLEEP_NoConfig_ Time_Capability	0x87	Time for the M-RX termination to be disabled and the LINE state to settle if inline configuration is not done. 1-15 SI. 3 Mbps: 1 SI = 3.33 us; 15 SI = 50 us 9 Mbps: 1 SI = 1.11 us; 15 SI = 16.67 us Value: ≤ 15 SI.

<table>
<tr><th>Attribute Name</th><th>AttributeID</th><th>Description</th></tr>
<tr><td>RX_Min_STALL_NoConfig_
Time_Capability</td><td>0x88</td><td>Time for the M-RX termination to be disabled and the LINE state to settle if inline configuration is not done. 1-255 SI.

<table><tr><th></th><th>1.25 Gbps</th><th>2.5 Gbps</th><th>5 Gbps</th></tr><tr><td>1 SI</td><td>8 ns</td><td>4 ns</td><td>2 ns</td></tr><tr><td>255 SI</td><td>2040 ns</td><td>1020 ns</td><td>510 ns</td></tr></table>
Value: ≤ 16 SI.</td></tr>
<tr><td>RX_Min_SAVE_Config_
Time_Capability</td><td>0x89</td><td>The greater of this time and remote RX_Min_SAVE_Config_Time_Capability is also honored on LINE-RESET exit when the M-TX is in SLEEP state.

1-250 (in 40 ns steps).

40 ns – 10 us.

~~Value: ≤ 125 (5 us).~~
Value: Implementation Specific.</td></tr>
<tr><td>RX_REF_CLOCK_
SHARED_Capability</td><td>0x8A</td><td>“0b”: No
“1b”: Yes

Value: Implementation specific.</td></tr>
<tr><td>RX_HS_G1_SYNC_LENGTH
_Capability</td><td>0x8B</td><td>[7:6]: Sync Range. Fine = 0, Coarse = 1.
[5:0]: Sync Length. 0 – 15.

If (Sync Range = FINE)
TSYNC = Sync Length (SI)
Else
TSYNC = 2Sync Length (SI)

Value: Implementation Specific.</td></tr>
</table>

Comment [VS374]: Mpcie_28August2014 Errata B6

<table>
<tr><th>Attribute Name</th><th>AttributeID</th><th>Description</th></tr>
<tr><td>RX_HS_G1__PREPARE_LENGTH_Capability</td><td>0x8C</td><td>0 to 15.

THS_PREPARE = HS_PREPARE_length * 2 (GEAR-1)

<table><tr><th></th><th>1.25 Gbps</th><th>2.5 Gbps</th><th>5 Gbps</th></tr><tr><td>1</td><td>1 SI
(8 ns)</td><td>2 SI
(8 ns)</td><td>4 SI
(8 ns)</td></tr><tr><td>15</td><td>15 SI
(120 ns)</td><td>30 SI
(120 ns)</td><td>60 SI
(120 ns)</td></tr></table>
Value: Implementation Specific.</td></tr>
<tr><td>RX_LS_PREPARE_LENGTH_Capability</td><td>0x8D</td><td>0 to 15.

TPWM_PREPARE = max (2LS_PREPARE_length + GEAR -7, 1)

<table><tr><th></th><th>3 Mbps</th><th>9 Mbps</th></tr><tr><td>1</td><td>1 SI (3.33 us)</td><td>1 SI (1.11 us)</td></tr><tr><td>15</td><td>512 SI
(1704.96 us)</td><td>512 SI
(568.32 us)</td></tr></table>
Value: = 1b.</td></tr>
<tr><td>RX_PWM_Burst_Closure_Length_Capability</td><td>0x8E</td><td>0 to 31 (SI).

~~Value: ≤ 7.~~
Value: ≤ 16.</td></tr>
<tr><td>RX_Min_ActivateTime_Capability</td><td>0x8F</td><td>1 to 9.

Value: 1.

M-PCIe does not use this capability. M-PCIe must use LINK_MIN_ACTIVATETIME_CAPABILITY as defined in Table 8-2.</td></tr>
</table>

Comment [VS375]: Mpcie_28August2014 Errata B4

Attribute Name	AttributeID	Description
RX_PHY_MajorMinor_Release _Capability	0x90	[7:4] Major version number. [3:0] Minor version number. Value: PHY Specific.
RX_PHY_Editorial_Release _Capability	0x91	1 to 99. Value: PHY Specific.
RX_Hibern8Time_Capability	0x92	When entering HIBERN8 from LS-MODE or HS-MODE, the protocol layer must not request exit before a minimum of THIBERN8, which is defined as the larger of local Tx_Hibern8Time_Capability and remote Rx_Hibern8Time_Capability. 1 to 128. Value: ≤ 100 us. This attribute is not used by M-PCIe. M-PCIe must use the LINK_HIBERN8_TIME_CAPABILITY attribute as defined in Table 8-2.
RX_PWM_G6_G7_SYNC_LEN GTH_Capability	0x93	Value: N/A. PCI Express Capability: No.
RX_HS_G2_SYNC_LENGTH _Capability	0x94	[7:6]: Sync Range. Fine = 0, Coarse = 1. [5:0]: Sync Length. 0 – 15. If (Sync Range = FINE) TSYNC = Sync Length (SI) Else TSYNC = 2Sync Length (SI) Value: Implementation Specific.

<table>
<tr><th>Attribute Name</th><th>AttributeID</th><th>Description</th></tr>
<tr><td>RX_HS_G3_SYNC_LENGTH_Capability</td><td>0x95</td><td>[7:6]: Sync Range. Fine = 0, Coarse = 1.
[5:0]: Sync Length. 0 – 15.

If (Sync Range = FINE)
TSYNC = Sync Length (SI)
Else
TSYNC = 2Sync Length (SI)

Value: Implementation Specific.</td></tr>
<tr><td>RX_HS_G2_PREPARE_LENGTH_Capability</td><td>0x96</td><td>0 to 15.

THS_PREPARE = HS_PREPARE_length * 2 (GEAR-1)

<table><tr><th></th><th>1.25 Gbps</th><th>2.5 Gbps</th><th>5 Gbps</th></tr><tr><td>1</td><td>1 SI
(8 ns)</td><td>2 SI
(8 ns)</td><td>4 SI
(8 ns)</td></tr><tr><td>15</td><td>15 SI
(120 ns)</td><td>30 SI
(120 ns)</td><td>60 SI
(120 ns)</td></tr></table>
Value: Implementation Specific.
PCI Express Capability: No.</td></tr>
<tr><td>RX_HS_G3_PREPARE_LENGTH_Capability</td><td>0x97</td><td>0 to 15.

THS_PREPARE = HS_PREPARE_length * 2 (GEAR-1)

<table><tr><th></th><th>1.25 Gbps</th><th>2.5 Gbps</th><th>5 Gbps</th></tr><tr><td>1</td><td>1 SI
(8 ns)</td><td>2 SI
(8 ns)</td><td>4 SI
(8 ns)</td></tr><tr><td>15</td><td>15 SI
(120 ns)</td><td>30 SI
(120 ns)</td><td>60 SI
(120 ns)</td></tr></table>
Value: Implementation Specific.</td></tr>
</table>

8.8.2. M-PHY Configuration Attributes

As noted in Annex D.2 in M-PHY specification, the M-TX Configuration attributes must be set appropriately to match the corresponding M-RX capability attribute values, the Link Initialization and configuration process as described in Section 8.2.1 must discover the capabilities and configure the attributes accordingly.

Some M-PHY attributes are configured based on implementation considerations, these attributes are **listed in Table 8-17.**

Table 8-17: M-TX Configuration Attributes

Attribute Name	AttributeID	Description	Reset Default
TX_HS_SlewRate	0x26	0 to 255	Implementation specific.
TX_LCC_Enable	0x2C	"0": No	TX_LCC_Enable must be disabled before entering PWM-Burst.
TX_DRIVER_POLARITY	0x2F	"0b": Normal "1b": Inverted	Implementation specific.

A

A. Isochronous Applications

A.1. Introduction

The design goal of isochronous mechanisms in PCI Express is to ensure that isochronous traffic receives its allocated bandwidth over a relevant time period while also preventing starvation of other non-isochronous traffic.

Furthermore, there may exist data traffic that requires a level of service falling in between what is required for bulk data traffic and isochronous data traffic. This type of traffic can be supported through the use of Port arbitration within Switches, the use of TC labels [1:7], and optional additional VC resources. Policies for assignment of TC labels and VC resources that are not isochronous-focused are outside the scope of the PCI Express specification.

Two paradigms of PCI Express communication are supported by the PCI Express isochronous mechanisms: Endpoint-to-Root-Complex communication model and peer-to-peer (Endpoint-to-Endpoint) communication model. In the Endpoint-to-Root-Complex communication model, the primary isochronous traffic is memory read and write requests to the Root Complex and read completions from the Root Complex. Figure A-1 shows an example of a simple system with both communication models. In the figure, devices A, B, called Requesters, are PCI Express Endpoints capable of issuing isochronous request transactions, while device C and Root Complex, called Completers, are capable of being the targets of isochronous request transactions. An Endpoint-to-Root-Complex communication is established between device A and the Root Complex, and a peer-to-peer communication is established between device B and device C. In the rest of this section, Requester and Completer will be used to make reference to PCI Express elements involved in transactions. The specific aspects of each communication model will be called out explicitly.

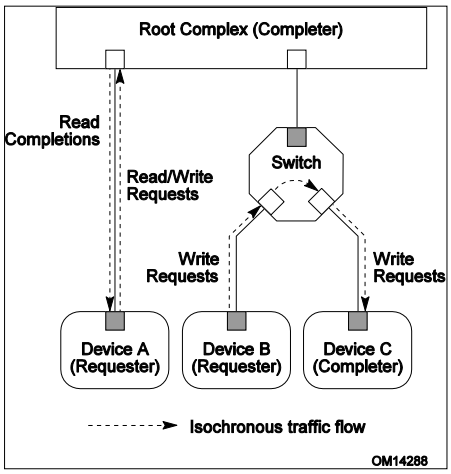


Figure A-1: An Example Showing Endpoint-to-Root-Complex and Peer-to-Peer Communication Models

Guaranteed bandwidth and deterministic latency require end-to-end configuration of fabric resources. If isochronous traffic is intermixed with non-isochronous traffic, it may not be possible to provide any guarantees/determinism as required by the application usage model. It is recommended that system software configure and assign fabric resources such that traffic intermix either does not occur or is such that the application usage model guarantees can be met. This can be accomplished by assigning dedicated VC resources and corresponding TC labels to the isochronous traffic flow(s) on a given path within the fabric.

Note that there may be one or more isochronous traffic flows per VC/TC label and it is up to system software to insure that the aggregation of these flows does not exceed the requisite bandwidth and latency requirements.

It is also possible for a fabric to support multiple isochronous traffic flows separated across multiple VC (a given flow cannot span multiple VC/TC labels).

In general, as long as the device can meet the isochronous bandwidth and latency requirements, there is nothing to preclude a single VC device from supporting isochronous traffic if multiple TC labels are supported to delineate such traffic from non-isochronous traffic within the fabric.

A.2. Isochronous Contract and Contract Parameters

In order to support isochronous data transfer with guaranteed bandwidth and deterministic latency, an isochronous contract must be established between a Requester/Completer pair and the PCI Express fabric. This contract must enforce both resource reservation and traffic regulation. Without such a contract, two basic problems, over-subscription and congestion, may occur as illustrated in Figure A-2. When interconnect bandwidth resources are over-subscribed, the increased latency may cause failure of isochronous service and starvation of non-isochronous services. Traffic congestion occurs when flow control credits are not returned possibly due to a higher than expected/provisioned packet injection rate. This may cause excessive service latencies for both isochronous traffic and non-isochronous traffic.

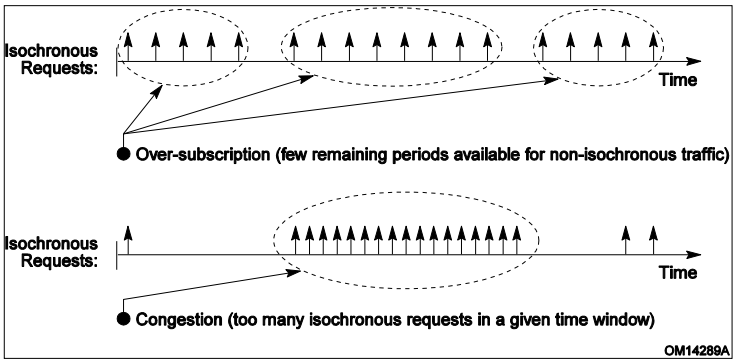


Figure A-2: Two Basic Bandwidth Resourcing Problems: Over-Subscription and Congestion

The isochronous transfer mechanism in this specification addresses these problems with traffic regulation including admission control and service discipline. Under a software managed admission control, a Requester must not issue isochronous transactions unless the required isochronous bandwidth and resource have been allocated. Specifically, the isochronous bandwidth is given by the following formula:

$$BW = \frac{N \cdot Y}{T}$$

The formula defines allocated bandwidth (BW) as a function of specified number (*N*) of transactions of a specified payload size (*Y*) within a specified time period (*T*). Another important parameter in the isochronous contract is latency. Based on the contract, isochronous transactions are completed within a specified latency (*L*). Once a Requester/Completer pair is admitted for isochronous communication, the bandwidth and latency are guaranteed to the Requester (a PCI Express Endpoint) by the Completer (Root Complex for Endpoint-to-Root-Complex communication and another PCI Express Endpoint for peer-to-peer communication) and by the PCI Express fabric components (Switches).

Specific service disciplines must be implemented by isochronous-capable PCI Express components. The service disciplines are imposed to PCI Express Switches and Completers in such a manner that the service of isochronous requests is subject to a specific service interval (*t*). This mechanism is used to provide the method of controlling when an isochronous packet injected by a Requester is serviced. Consequently, isochronous traffic is policed in such manner that only packets that can be injected into the fabric in compliance with the isochronous contract are allowed to make immediate progress and start being serviced by the PCI Express fabric. A non-compliant Requester that tries to inject more isochronous transactions than what was being allowed by the contract is prevented from doing so by the flow-control mechanism thereby allowing compliant Requesters to correctly operate independent of non-compliant Requesters.

In the Endpoint-to-Root-Complex model, since the aggregated isochronous traffic is eventually limited by the host memory subsystem's bandwidth capabilities, isochronous read requests, and write requests (and Messages) are budgeted together. A Requester may divide the isochronous bandwidth between read requests and write requests as appropriate.

A.2.1. Isochronous Time Period and Isochronous Virtual Timeslot

The PCI Express isochronous time period (*T*) is uniformly divided into units of virtual timeslots (*t*). To provide precise isochronous bandwidth distribution only one isochronous request packet is allowed per virtual timeslot. The virtual timeslot supported by a PCI Express component is reported through the Reference Clock field in the Virtual Channel Capability structure or the Multi-Function Virtual Channel Capability structure. When Reference Clock = 00b, duration of a virtual timeslot *t* is 100 ns. Duration of isochronous time period *T* depends on the number of phases of the supported time-based WRR Port arbitration table size. When the time-based WRR Port Arbitration Table size equals to 128, there are 128 virtual timeslots (*t*) in an isochronous time period, i.e. T = 12.8 µs.

Note that isochronous period *T* as well as virtual timeslots *t* do not need to be aligned and synchronized among different PCI Express isochronous devices, i.e., the notion of {*T, t*} is local to each individual isochronous device.

A.2.2. Isochronous Payload Size

The payload size (*Y*) for isochronous transactions must not exceed Max_Payload_Size (see Section 7.8.4). After configuration, the Max_Payload_Size is set and fixed for each path that supports isochronous service with a value required to meet isochronous latency. The fixed Max_Payload_Size value is used for isochronous bandwidth budgeting regardless of the actual size of data payload associated with isochronous transactions. For isochronous bandwidth budgeting, we have

$$Y = \text{Max_Payload_Size}$$

A transaction with partial writes is treated as a normally accounted transaction. A Completer must account for partial writes as part of bandwidth assignment (for worst case servicing time).

A.2.3. Isochronous Bandwidth Allocation

Given *T*, *t* and *Y*, the maximum virtual timeslots within a time period is

$$N_{\max} = \frac{T}{t}$$

and the maximum specifiable isochronous bandwidth is

$$BW_{\max} = \frac{Y}{t}$$

The granularity with which isochronous bandwidth can be allocated is defined as:

$$BW_{granularity} = \frac{Y}{T}$$

Given *T* and *t* at 12.8 µs and 100 ns, respectively, N_{max} is 128. As shown in Table A-1, BW_{max} and $BW_{granularity}$ are functions of the isochronous payload size *Y*.

Table A-1: Isochronous Bandwidth Ranges and Granularities

Y (bytes)	128	256	512	1024
BW_{max} (MB/s)	1280	2560	5120	10240
$BW_{granularity}$ (MB/s)	10	20	40	80

Similar to bandwidth budgeting, isochronous service disciplines including arbitration schemes are based on counting requests (not the sizes of those requests. Therefore, assigning isochronous bandwidth BW_{link} to a PCI Express Link is equivalent to assigning N_{link} virtual timeslots per isochronous period, where N_{link} is given by

$$N_{link} = \frac{BW_{link}}{BW_{granularity}}$$

A Switch Port serving as an Egress Port (or an RCRB serving as a "virtual" Egress Port) for an isochronous traffic, the N_{max} virtual timeslots within *T* are represented by the time-based WRR Port Arbitration Table in the PCI Express Virtual Channel Capability structure detailed in Section 7.11.

The table consists of N_{max} entries. An entry in the table represents one virtual timeslot in the isochronous time period. When a table entry is given a value of PN, it means that the timeslot is assigned to an Ingress Port (in respect to the isochronous traffic targeting the Egress Port) designated by a Port Number of PN. Therefore, N_{link} virtual timeslots are assigned to the Ingress Port when there are N_{link} entries in the table with value of PN. The Egress Port may admit one isochronous request transaction from the Ingress Port for further service only when the table entry reached by the Egress Port's isochronous time ticker (that increments by 1 every t time and wraps around when reaching T) is set to PN. Even if there are outstanding isochronous requests ready in the Ingress Port, they will not be served until next round of time-based WRR arbitration. In this manner, the time-based Port Arbitration Table serves for both isochronous bandwidth assignment and isochronous traffic regulation.

For an Endpoint serving as a Requester or a Completer, isochronous bandwidth allocation is accomplished through negotiation between system software and device driver, which is outside of the scope of this specification.

A.2.4. Isochronous Transaction Latency

Transaction latency is composed of the latency through the PCI Express fabric and the latency contributed by the Completer. Isochronous transaction latency is defined for each transaction and measured in units of virtual timeslot t.

- The *read latency* is defined as the round-trip latency. This is the delay from the time when the device submits a memory read request packet to its Transaction Layer (Transmit side) to the time when the corresponding read completion arrives at the device's Transaction Layer (Receive side).
- The *write latency* is defined as the delay from the time when the Requester posts a memory write request to its PCI Express Transaction Layer (Transmit side) to the time when the data write becomes globally visible within the memory subsystem of the Completer. A write to memory reaches the point of global visibility when all agents accessing that memory address get the updated data.

When the upper bound and the lower bound of isochronous transaction latency are provided, the size of isochronous data buffers in a Requester can be determined. For most of common platforms, the minimum isochronous transaction latency is much smaller than the maximum. As a conservative measure, the minimum isochronous transaction latency is assumed to be zero; only guidelines on measuring the maximum isochronous transaction latency are provided here.

For a Requester, the maximum isochronous (read or write) transaction latency (L) can be accounted as the following:

$$L = L_{Fabric} + L_{Completer},$$

where L_{Fabric} is the maximum latency of the PCI Express fabric and $L_{Completer}$ is the maximum latency of the Completer.

L_{Fabric} which applies to both read and write transactions, depends on the topology, latency across each PCI Express Link, and the arbitration point in the path between the Requester to the Completer. The latency on a PCI Express Link depends on pipeline delays, width and operational

frequency of the Link, transmission of electrical signals across the medium, wake up latency from low power states, and delays caused by Data Link Layer Retry.

A restriction on the PCI Express topology may be imposed for each targeted platform in order to provide a practically meaningful guideline for L_{Fabric}. The values of L_{Fabric} should be reasonable and serve as practical upper limits under normal operating conditions.

The value of $L_{Completer}$ depends on the memory technology, memory configuration, and the arbitration policies in the Completer that comprehend PCI Express isochronous traffic. The target value for $L_{Completer}$ should provide enough headroom to allow for implementation tradeoffs.

Definitions of read and write transaction latencies for a Completer are different:

- Read transaction latency for the Completer is defined as the delay from the time a memory read transaction is available at the Receiver end of a PCI Express Port in the Completer to the time the corresponding read completion transaction is posted to the transmission end of the PCI Express Port.
- Write transaction latency is defined as the delay from the time a memory write transaction is available at the Receiver end of a PCI Express Port in the Completer to the time that the transmitted data is globally visible.

All of the isochronous transaction latencies defined above are based on the assumption that the Requester injects isochronous transactions uniformly. According to an isochronous contract of $\{N, T, t\}$, the uniform traffic injection is defined such that up to N transactions are evenly distributed over the isochronous period T based on a ticker granularity of virtual timeslot t. For a Requester with non-uniform isochronous transaction injection, the Requester is responsible of accounting for any additional delay due to the deviation of its injection pattern from a uniform injection pattern.

A.2.5. An Example Illustrating Isochronous Parameters

Figure A-3 illustrates the key isochronous parameters using a simplified example with $T = 20t$ and $L = 22t$. A Requester has reserved isochronous bandwidth of four transactions per T. The device shares the allocated isochronous bandwidths for both read requests and write requests. As shown, during one isochronous time period, the Requester issues two read requests and two write requests. All requests are completed within the designated transaction latency L. Also shown in the figure, there is no time dependency between the service time of write requests and the arrival time of read completions.

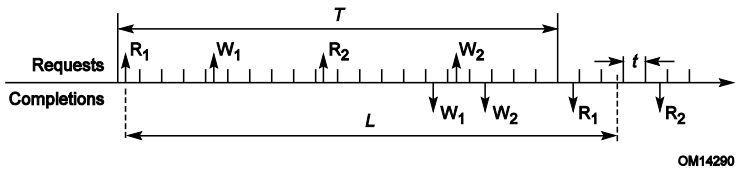


Figure A-3: A Simplified Example Illustrating PCI Express Isochronous Parameters

A.3. Isochronous Transaction Rules

Isochronous transactions follow the same rules as described in Chapter 2. In order to assist the Completer to meet latency requirements, the following additional rules further illustrate and clarify the proper behavior of isochronous transactions:

- The value in the Length field of requests must never exceed Max_Payload_Size.

A.4. Transaction Ordering

In general, isochronous transactions follow the ordering rules described in Section 2.4. The following ordering rule further illustrates and clarifies the proper behavior of isochronous transactions:

- There are no ordering guarantees between any isochronous and non-isochronous transactions because the traffic has been segregated into distinct VC resources.
- Isochronous write requests are serviced on any PCI Express Link in strictly the same order as isochronous write requests are posted.
- Switches must allow isochronous posted requests to pass isochronous read completions.

A.5. Isochronous Data Coherency

Cache coherency for isochronous transactions is an operating system software and Root Complex hardware issue. PCI Express provides the necessary mechanism to control Root Complex behavior in terms of enforcing hardware cache coherency on a per transaction basis.

For platforms where snoop latency in a Root Complex is either unbounded or can be excessively large, in order to meet tight maximum isochronous transaction latency $L_{Completer}$, or more precisely $L_{Root_Complex}$, all isochronous transactions should have the No Snoop Attribute bit set.

A Root Complex must report the Root Complex's capability to the system software by setting the Reject Snoop Transactions field in the VC Resource Capability register (for any VC resource capable of supporting isochronous traffic) in its RCRB. Based on whether or not a Root Complex is capable of providing hardware enforced cache coherency for isochronous traffic while still meeting isochronous latency target, system software can then inform the device driver of Endpoints to set or unset the No Snoop Attribute bit for isochronous transactions.

Note that cache coherency considerations for isochronous traffic do not apply to peer-to-peer communication.

A.6. Flow Control

Completers and PCI Express fabric components should implement proper sizing of buffers such that under normal operating conditions, no backpressure due to flow control should be applied to isochronous traffic injected uniformly by a Requester. For Requesters that are compliant to the isochronous contract, but have bursty injection behavior, Switches and Completers may apply flow control backpressure as long as the admitted isochronous traffic is uniform and compliant to the isochronous contract. Under abnormal conditions when isochronous traffic jitter becomes significant or when isochronous traffic is oversubscribed due to excessive Data Link Layer Retry, flow control provides a natural mechanism to ensure functional correctness.

A.7. Considerations for Bandwidth Allocation

A.7.1. Isochronous Bandwidth of PCI Express Links

Isochronous bandwidth budgeting for PCI Express Links can be derived based on Link parameters such as isochronous payload size and the speed and width of the Link.

Isochronous bandwidth allocation for a PCI Express Link should be limited to certain percentage of the maximum effective Link bandwidth in order to leave sufficient bandwidth for non-isochronous traffic and to account for temporary Link bandwidth reduction due to retries. Link utilization is counted based on the actual cycles consumed on the physical PCI Express Link. The maximum number of virtual slots allowed per Link (N_{link}) depends on the isochronous packet payload size and the speed and width of the Link.

As isochronous bandwidth allocation on a PCI Express Link is based on number of requests N_{link} per isochronous period. There is no distinction between read requests and write requests in budgeting isochronous bandwidth on a PCI Express Link.

A.7.2. Isochronous Bandwidth of Endpoints

For peer-to-peer communication, the device driver is responsible for reporting to system software if the device is capable of being a Completer for isochronous transactions. In addition, the driver must report the device's isochronous bandwidth capability. The specifics of the report mechanism are outside the scope of this specification.

A.7.3. Isochronous Bandwidth of Switches

Allocation of isochronous bandwidth for a Switch must consider the capacity and utilization of PCI Express Links associated with the Ingress Port and the Egress Port of the Switch that connect the Requester and the Completer, respectively. The lowest common denominator of the two determines if a requested isochronous bandwidth can be supported.

A.7.4. Isochronous Bandwidth of Root Complex

Isochronous bandwidth of Root Complex is reported to the software through its RCRB structure. Specifically, the Maximum Time Slots field of the VC Resource Capability register in VC Capability structure indicates the total isochronous bandwidth shared by the Root Ports associated with the RCRB. Details of the platform budgeting for available isochronous bandwidth within a Root Complex are outside of the scope of this specification.

A.8. Considerations for PCI Express Components

A.8.1. An Endpoint as a Requester

Before an Endpoint as a Requester can start issuing isochronous request transactions, the following configuration steps must be performed by software:

- Configuration of at least one VC resource capable of supporting isochronous communication and assignment of at least one TC label.
- Enablement of this VC resource.

When the Requester uniformly injects isochronous requests, the Receive Port, either a Switch Port or a Root Port, should issue Flow Control credits back promptly such that no backpressure should be applied to the associated VC. This type of Requester may size its buffer based on the PCI Express fabric latency L_{Fabric} plus the Completer's latency $L_{Completer}$.

When isochronous transactions are injected non-uniformly, either some transactions experience longer PCI Express fabric delay or the Requester gets back-pressured on the associated VC. This type of Requester must size its buffer to account for the deviation of its injection pattern from uniformity.

A.8.2. An Endpoint as a Completer

An Endpoint may serve as a Completer for isochronous peer-to-peer communication. Before an Endpoint starts serving isochronous transactions, system software must identify/configure a VC resource capable of supporting isochronous traffic and assigned a corresponding TC label.

An Endpoint Completer must observe the maximum isochronous transaction latency ($L_{Completer}$). An Endpoint Completer does not have to regulate isochronous request traffic if attached to a Switch since Switches implement traffic regulation. However, an Endpoint Completer must size its internal buffer such that no backpressure should be applied to the corresponding VC.

A.8.3. Switches

A Switch may have multiple ports capable of supporting isochronous transactions. Before a Switch starts serving isochronous transactions for a Port, the software must perform the following configuration steps:

- Configuration/enablement of at least one VC resource capable of supporting isochronous communication.
- Configuration of the Port as an Ingress Port:
 - Configuration (or reconfiguration if the associated VC of the Egress Port is already enabled) of the time-based WRR Port Arbitration Table of the targeting Egress Port to include N_{link} entries set to the Ingress Port's Port Number. Here N_{link} is the isochronous allocation for the Ingress Port.
 - Enabling the targeting Egress Port to load newly programmed Port Arbitration Table.
- Configuration of the Port as an Egress Port:
 - Configuration of each VC's Port Arbitration Table with number of entries set according to the assigned isochronous bandwidth for all Ingress Ports.
 - Select proper VC Arbitration, e.g., as strict-priority based VC Arbitration.
 - If required, configuration of the Port's VC Arbitration Table with large weights assigned accordingly to each associated VC.

Each VC associated with isochronous traffic may be served as the highest priority in arbitrating for the shared PCI Express Link resource at an Egress Port. This is comprehended by a Switch's internal arbitration scheme.

In addition, a Switch Port may use "just in time" scheduling mechanism to reduce VC arbitration latency. Instead of pipelining non-isochronous Transport Layer packets to the Data Link Layer of the Egress Port in a manner that Data Link Layer transmit buffer becomes saturated, the Switch Port may hold off scheduling of a new non-isochronous packet to the Data Link Layer as long as it is possible without incurring unnecessary Link idle time.

When a VC configured to support isochronous traffic is enabled for a Switch Port (ingress) that is connected to a Requester, the Switch must enforce proper traffic regulation to ensure that isochronous traffic from the Port conforms to this specification. With such enforcement, normal isochronous transactions from compliant Requesters will not be impacted by ill behavior of any non-compliant Requester.

The above isochronous traffic regulation mechanism only applies to request transactions but not to completion transactions. When Endpoint-to-Root-Complex and peer-to-peer communications co-exist in a Switch, an Egress Port may mix isochronous write requests and read completions in the same direction. In the case of contention, the Egress Port must allow write requests to pass read completions to ensure the Switch meets latency requirement for isochronous requests.

A.8.4. Root Complex

A Root Complex may have multiple Root Ports capable of supporting isochronous transactions. Before a Root Complex starts serving isochronous transactions for a Root Port, the Port must be configured by software to enable VC to support isochronous traffic using the following configuration steps:

- ❑ Configuration of at least one VC resource capable of supporting isochronous communication and assignment of at least one TC label.
- ❑ Configuration of the Root Port as an Ingress Port:
 - Configuration (or reconfiguration if the associated VC in RCRB is already enabled) of the time-based WRR Port Arbitration Table of the targeting RCRB to include N_{link} entries set to the Ingress Port's Port Number. Here N_{link} is the isochronous allocation for the Port.
 - Enabling the targeting RCRB to load newly programmed Port Arbitration Table.
- ❑ Configuration of the Root Port as an Egress Port:
 - If supported, configuration of the Root Port's VC Arbitration Table with large weights assigned to the associated VC.
 - If the Root Complex supports peer-to-peer traffic between Root Ports, configuration of the Root Port's Port Arbitration Table number of entries is set according to the assigned isochronous bandwidth for all Ingress Ports.

A Root Complex must observe the maximum isochronous transaction latency ($L_{Completer}$ or more precisely $L_{Root_Complex}$) that applies to all the Root Ports in the Root Complex. How a Root Complex schedules memory cycles for PCI Express isochronous transactions and other memory transactions is outside of the scope of this specification as long as $L_{Root_Complex}$ is met for PCI Express isochronous transactions.

When a VC is enabled to support isochronous traffic for a Root Port, the Root Complex must enforce proper traffic regulation to ensure that isochronous traffic from the Root Port conforms to this specification. With such enforcement, normal isochronous transactions from compliant Requesters will not be impacted by ill behavior of any non-compliant Requesters. Isochronous traffic regulation is implemented using the time-based Port Arbitration Table in RCRB.

Root Complex may perform the following operations for invalid isochronous transactions:

- ❑ Return partial completions for read requests with the value in the Length field exceeding Max_Payload_Size.

B

B. Symbol Encoding

Table B-1 shows the byte-to-Symbol encodings for data characters. Table B-2 shows the Symbol encodings for the Special Symbols used for TLP/DLLP Framing and for interface management. RD- and RD+ refer to the Running Disparity of the Symbol sequence on a per-Lane basis.

Table B-1: 8b/10b Data Symbol Codes

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D0.0	00	000 00000	100111 0100	011000 1011
D1.0	01	000 00001	011101 0100	100010 1011
D2.0	02	000 00010	101101 0100	010010 1011
D3.0	03	000 00011	110001 1011	110001 0100
D4.0	04	000 00100	110101 0100	001010 1011
D5.0	05	000 00101	101001 1011	101001 0100
D6.0	06	000 00110	011001 1011	011001 0100
D7.0	07	000 00111	111000 1011	000111 0100
D8.0	08	000 01000	111001 0100	000110 1011
D9.0	09	000 01001	100101 1011	100101 0100
D10.0	0A	000 01010	010101 1011	010101 0100
D11.0	0B	000 01011	110100 1011	110100 0100
D12.0	0C	000 01100	001101 1011	001101 0100
D13.0	0D	000 01101	101100 1011	101100 0100
D14.0	0E	000 01110	011100 1011	011100 0100
D15.0	0F	000 01111	010111 0100	101000 1011
D16.0	10	000 10000	011011 0100	100100 1011
D17.0	11	000 10001	100011 1011	100011 0100
D18.0	12	000 10010	010011 1011	010011 0100
D19.0	13	000 10011	110010 1011	110010 0100
D20.0	14	000 10100	001011 1011	001011 0100
D21.0	15	000 10101	101010 1011	101010 0100
D22.0	16	000 10110	011010 1011	011010 0100

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D23.0	17	000 10111	111010 0100	000101 1011
D24.0	18	000 11000	110011 0100	001100 1011
D25.0	19	000 11001	100110 1011	100110 0100
D26.0	1A	000 11010	010110 1011	010110 0100
D27.0	1B	000 11011	110110 0100	001001 1011
D28.0	1C	000 11100	001110 1011	001110 0100
D29.0	1D	000 11101	101110 0100	010001 1011
D30.0	1E	000 11110	011110 0100	100001 1011
D31.0	1F	000 11111	101011 0100	010100 1011
D0.1	20	001 00000	100111 1001	011000 1001
D1.1	21	001 00001	011101 1001	100010 1001
D2.1	22	001 00010	101101 1001	010010 1001
D3.1	23	001 00011	110001 1001	110001 1001
D4.1	24	001 00100	110101 1001	001010 1001
D5.1	25	001 00101	101001 1001	101001 1001
D6.1	26	001 00110	011001 1001	011001 1001
D7.1	27	001 00111	111000 1001	000111 1001
D8.1	28	001 01000	111001 1001	000110 1001
D9.1	29	001 01001	100101 1001	100101 1001
D10.1	2A	001 01010	010101 1001	010101 1001
D11.1	2B	001 01011	110100 1001	110100 1001
D12.1	2C	001 01100	001101 1001	001101 1001
D13.1	2D	001 01101	101100 1001	101100 1001
D14.1	2E	001 01110	011100 1001	011100 1001
D15.1	2F	001 01111	010111 1001	101000 1001
D16.1	30	001 10000	011011 1001	100100 1001
D17.1	31	001 10001	100011 1001	100011 1001
D18.1	32	001 10010	010011 1001	010011 1001
D19.1	33	001 10011	110010 1001	110010 1001
D20.1	34	001 10100	001011 1001	001011 1001
D21.1	35	001 10101	101010 1001	101010 1001
D22.1	36	001 10110	011010 1001	011010 1001
D23.1	37	001 10111	111010 1001	000101 1001

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D24.1	38	001 11000	110011 1001	001100 1001
D25.1	39	001 11001	100110 1001	100110 1001
D26.1	3A	001 11010	010110 1001	010110 1001
D27.1	3B	001 11011	110110 1001	001001 1001
D28.1	3C	001 11100	001110 1001	001110 1001
D29.1	3D	001 11101	101110 1001	010001 1001
D30.1	3E	001 11110	011110 1001	100001 1001
D31.1	3F	001 11111	101011 1001	010100 1001
D0.2	40	010 00000	100111 0101	011000 0101
D1.2	41	010 00001	011101 0101	100010 0101
D2.2	42	010 00010	101101 0101	010010 0101
D3.2	43	010 00011	110001 0101	110001 0101
D4.2	44	010 00100	110101 0101	001010 0101
D5.2	45	010 00101	101001 0101	101001 0101
D6.2	46	010 00110	011001 0101	011001 0101
D7.2	47	010 00111	111000 0101	000111 0101
D8.2	48	010 01000	111001 0101	000110 0101
D9.2	49	010 01001	100101 0101	100101 0101
D10.2	4A	010 01010	010101 0101	010101 0101
D11.2	4B	010 01011	110100 0101	110100 0101
D12.2	4C	010 01100	001101 0101	001101 0101
D13.2	4D	010 01101	101100 0101	101100 0101
D14.2	4E	010 01110	011100 0101	011100 0101
D15.2	4F	010 01111	010111 0101	101000 0101
D16.2	50	010 10000	011011 0101	100100 0101
D17.2	51	010 10001	100011 0101	100011 0101
D18.2	52	010 10010	010011 0101	010011 0101
D19.2	53	010 10011	110010 0101	110010 0101
D20.2	54	010 10100	001011 0101	001011 0101
D21.2	55	010 10101	101010 0101	101010 0101
D22.2	56	010 10110	011010 0101	011010 0101
D23.2	57	010 10111	111010 0101	000101 0101
D24.2	58	010 11000	110011 0101	001100 0101

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D25.2	59	010 11001	100110 0101	100110 0101
D26.2	5A	010 11010	010110 0101	010110 0101
D27.2	5B	010 11011	110110 0101	001001 0101
D28.2	5C	010 11100	001110 0101	001110 0101
D29.2	5D	010 11101	101110 0101	010001 0101
D30.2	5E	010 11110	011110 0101	100001 0101
D31.2	5F	010 11111	101011 0101	010100 0101
D0.3	60	011 00000	100111 0011	011000 1100
D1.3	61	011 00001	011101 0011	100010 1100
D2.3	62	011 00010	101101 0011	010010 1100
D3.3	63	011 00011	110001 1100	110001 0011
D4.3	64	011 00100	110101 0011	001010 1100
D5.3	65	011 00101	101001 1100	101001 0011
D6.3	66	011 00110	011001 1100	011001 0011
D7.3	67	011 00111	111000 1100	000111 0011
D8.3	68	011 01000	111001 0011	000110 1100
D9.3	69	011 01001	100101 1100	100101 0011
D10.3	6A	011 01010	010101 1100	010101 0011
D11.3	6B	011 01011	110100 1100	110100 0011
D12.3	6C	011 01100	001101 1100	001101 0011
D13.3	6D	011 01101	101100 1100	101100 0011
D14.3	6E	011 01110	011100 1100	011100 0011
D15.3	6F	011 01111	010111 0011	101000 1100
D16.3	70	011 10000	011011 0011	100100 1100
D17.3	71	011 10001	100011 1100	100011 0011
D18.3	72	011 10010	010011 1100	010011 0011
D19.3	73	011 10011	110010 1100	110010 0011
D20.3	74	011 10100	001011 1100	001011 0011
D21.3	75	011 10101	101010 1100	101010 0011
D22.3	76	011 10110	011010 1100	011010 0011
D23.3	77	011 10111	111010 0011	000101 1100
D24.3	78	011 11000	110011 0011	001100 1100
D25.3	79	011 11001	100110 1100	100110 0011

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D26.3	7A	011 11010	010110 1100	010110 0011
D27.3	7B	011 11011	110110 0011	001001 1100
D28.3	7C	011 11100	001110 1100	001110 0011
D29.3	7D	011 11101	101110 0011	010001 1100
D30.3	7E	011 11110	011110 0011	100001 1100
D31.3	7F	011 11111	101011 0011	010100 1100
D0.4	80	100 00000	100111 0010	011000 1101
D1.4	81	100 00001	011101 0010	100010 1101
D2.4	82	100 00010	101101 0010	010010 1101
D3.4	83	100 00011	110001 1101	110001 0010
D4.4	84	100 00100	110101 0010	001010 1101
D5.4	85	100 00101	101001 1101	101001 0010
D6.4	86	100 00110	011001 1101	011001 0010
D7.4	87	100 00111	111000 1101	000111 0010
D8.4	88	100 01000	111001 0010	000110 1101
D9.4	89	100 01001	100101 1101	100101 0010
D10.4	8A	100 01010	010101 1101	010101 0010
D11.4	8B	100 01011	110100 1101	110100 0010
D12.4	8C	100 01100	001101 1101	001101 0010
D13.4	8D	100 01101	101100 1101	101100 0010
D14.4	8E	100 01110	011100 1101	011100 0010
D15.4	8F	100 01111	010111 0010	101000 1101
D16.4	90	100 10000	011011 0010	100100 1101
D17.4	91	100 10001	100011 1101	100011 0010
D18.4	92	100 10010	010011 1101	010011 0010
D19.4	93	100 10011	110010 1101	110010 0010
D20.4	94	100 10100	001011 1101	001011 0010
D21.4	95	100 10101	101010 1101	101010 0010
D22.4	96	100 10110	011010 1101	011010 0010
D23.4	97	100 10111	111010 0010	000101 1101
D24.4	98	100 11000	110011 0010	001100 1101
D25.4	99	100 11001	100110 1101	100110 0010
D26.4	9A	100 11010	010110 1101	010110 0010

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D27.4	9B	100 11011	110110 0010	001001 1101
D28.4	9C	100 11100	001110 1101	001110 0010
D29.4	9D	100 11101	101110 0010	010001 1101
D30.4	9E	100 11110	011110 0010	100001 1101
D31.4	9F	100 11111	101011 0010	010100 1101
D0.5	A0	101 00000	100111 1010	011000 1010
D1.5	A1	101 00001	011101 1010	100010 1010
D2.5	A2	101 00010	101101 1010	010010 1010
D3.5	A3	101 00011	110001 1010	110001 1010
D4.5	A4	101 00100	110101 1010	001010 1010
D5.5	A5	101 00101	101001 1010	101001 1010
D6.5	A6	101 00110	011001 1010	011001 1010
D7.5	A7	101 00111	111000 1010	000111 1010
D8.5	A8	101 01000	111001 1010	000110 1010
D9.5	A9	101 01001	100101 1010	100101 1010
D10.5	AA	101 01010	010101 1010	010101 1010
D11.5	AB	101 01011	110100 1010	110100 1010
D12.5	AC	101 01100	001101 1010	001101 1010
D13.5	AD	101 01101	101100 1010	101100 1010
D14.5	AE	101 01110	011100 1010	011100 1010
D15.5	AF	101 01111	010111 1010	101000 1010
D16.5	B0	101 10000	011011 1010	100100 1010
D17.5	B1	101 10001	100011 1010	100011 1010
D18.5	B2	101 10010	010011 1010	010011 1010
D19.5	B3	101 10011	110010 1010	110010 1010
D20.5	B4	101 10100	001011 1010	001011 1010
D21.5	B5	101 10101	101010 1010	101010 1010
D22.5	B6	101 10110	011010 1010	011010 1010
D23.5	B7	101 10111	111010 1010	000101 1010
D24.5	B8	101 11000	110011 1010	001100 1010
D25.5	B9	101 11001	100110 1010	100110 1010
D26.5	BA	101 11010	010110 1010	010110 1010
D27.5	BB	101 11011	110110 1010	001001 1010

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D28.5	BC	101 11100	001110 1010	001110 1010
D29.5	BD	101 11101	101110 1010	010001 1010
D30.5	BE	101 11110	011110 1010	100001 1010
D31.5	BF	101 11111	101011 1010	010100 1010
D0.6	C0	110 00000	100111 0110	011000 0110
D1.6	C1	110 00001	011101 0110	100010 0110
D2.6	C2	110 00010	101101 0110	010010 0110
D3.6	C3	110 00011	110001 0110	110001 0110
D4.6	C4	110 00100	110101 0110	001010 0110
D5.6	C5	110 00101	101001 0110	101001 0110
D6.6	C6	110 00110	011001 0110	011001 0110
D7.6	C7	110 00111	111000 0110	000111 0110
D8.6	C8	110 01000	111001 0110	000110 0110
D9.6	C9	110 01001	100101 0110	100101 0110
D10.6	CA	110 01010	010101 0110	010101 0110
D11.6	CB	110 01011	110100 0110	110100 0110
D12.6	CC	110 01100	001101 0110	001101 0110
D13.6	CD	110 01101	101100 0110	101100 0110
D14.6	CE	110 01110	011100 0110	011100 0110
D15.6	CF	110 01111	010111 0110	101000 0110
D16.6	D0	110 10000	011011 0110	100100 0110
D17.6	D1	110 10001	100011 0110	100011 0110
D18.6	D2	110 10010	010011 0110	010011 0110
D19.6	D3	110 10011	110010 0110	110010 0110
D20.6	D4	110 10100	001011 0110	001011 0110
D21.6	D5	110 10101	101010 0110	101010 0110
D22.6	D6	110 10110	011010 0110	011010 0110
D23.6	D7	110 10111	111010 0110	000101 0110
D24.6	D8	110 11000	110011 0110	001100 0110
D25.6	D9	110 11001	100110 0110	100110 0110
D26.6	DA	110 11010	010110 0110	010110 0110
D27.6	DB	110 11011	110110 0110	001001 0110
D28.6	DC	110 11100	001110 0110	001110 0110

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D29.6	DD	110 11101	101110 0110	010001 0110
D30.6	DE	110 11110	011110 0110	100001 0110
D31.6	DF	110 11111	101011 0110	010100 0110
D0.7	E0	111 00000	100111 0001	011000 1110
D1.7	E1	111 00001	011101 0001	100010 1110
D2.7	E2	111 00010	101101 0001	010010 1110
D3.7	E3	111 00011	110001 1110	110001 0001
D4.7	E4	111 00100	110101 0001	001010 1110
D5.7	E5	111 00101	101001 1110	101001 0001
D6.7	E6	111 00110	011001 1110	011001 0001
D7.7	E7	111 00111	111000 1110	000111 0001
D8.7	E8	111 01000	111001 0001	000110 1110
D9.7	E9	111 01001	100101 1110	100101 0001
D10.7	EA	111 01010	010101 1110	010101 0001
D11.7	EB	111 01011	110100 1110	110100 1000
D12.7	EC	111 01100	001101 1110	001101 0001
D13.7	ED	111 01101	101100 1110	101100 1000
D14.7	EE	111 01110	011100 1110	011100 1000
D15.7	EF	111 01111	010111 0001	101000 1110
D16.7	F0	111 10000	011011 0001	100100 1110
D17.7	F1	111 10001	100011 0111	100011 0001
D18.7	F2	111 10010	010011 0111	010011 0001
D19.7	F3	111 10011	110010 1110	110010 0001
D20.7	F4	111 10100	001011 0111	001011 0001
D21.7	F5	111 10101	101010 1110	101010 0001
D22.7	F6	111 10110	011010 1110	011010 0001
D23.7	F7	111 10111	111010 0001	000101 1110
D24.7	F8	111 11000	110011 0001	001100 1110
D25.7	F9	111 11001	100110 1110	100110 0001
D26.7	FA	111 11010	010110 1110	010110 0001
D27.7	FB	111 11011	110110 0001	001001 1110
D28.7	FC	111 11100	001110 1110	001110 0001
D29.7	FD	111 11101	101110 0001	010001 1110

Data Byte Name	Data Byte Value (hex)	Bits HGF EDCBA (binary)	Current RD- abcdei fghj(binary)	Current RD+ abcdei fghj (binary)
D30.7	FE	111 11110	011110 0001	100001 1110
D31.7	FF	111 11111	101011 0001	010100 1110

Table B-2: 8b/10b Special Character Symbol Codes

Data Byte Name	Data Byte Value	Bits HGF EDCBA	Current RD - abcdei fghj	Current RD + abcdei fghj
K28.0	1C	000 11100	001111 0100	110000 1011
K28.1	3C	001 11100	001111 1001	110000 0110
K28.2	5C	010 11100	001111 0101	110000 1010
K28.3	7C	011 11100	001111 0011	110000 1100
K28.4	9C	100 11100	001111 0010	110000 1101
K28.5	BC	101 11100	001111 1010	110000 0101
K28.6	DC	110 11100	001111 0110	110000 1001
K28.7	FC	111 11100	001111 1000	110000 0111
K23.7	F7	111 10111	111010 1000	000101 0111
K27.7	FB	111 11011	110110 1000	001001 0111
K29.7	FD	111 11101	101110 1000	010001 0111
K30.7	FE	111 11110	011110 1000	100001 0111

C

C. Physical Layer Appendix

C.1. 8b/10b Data Scrambling Example

The following subroutines encode and decode an 8-bit value contained in "inbyte" with the LFSR. This is presented as one example only; there are many ways to obtain the proper output. This example demonstrates how to advance the LFSR eight times in one operation and how to XOR the data in one operation. Many other implementations are possible but they must all produce the same output as that shown here.

The following algorithm uses the "C" programming language conventions, where "<<" and ">>" represent the shift left and shift right operators, ">" is the compare greater than operator, and "^" is the exclusive or operator, and "&" is the logical "AND" operator.

```
/*
   this routine implements the serial descrambling algorithm in parallel form
   for the LSFR polynomial:  x^16+x^5+x^4+x^3+1
   this advances the LSFR 8 bits every time it is called
   this requires fewer than 25 xor gates to implement (with a static register)

   The XOR required to advance 8 bits/clock is:
bit   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15

      8   9  10  11  12  13  14  15   0   1   2   3   4   5   6   7
                  8   9  10  11  12  13  14  15
                      8   9  10  11  12  13  14  15
                          8   9  10  11  12  13  14  15

   The serial data is just the reverse of the upper byte:
bit   0   1   2   3   4   5   6   7
     15  14  13  12  11  10   9   8
*/

int scramble_byte(int inbyte)
{

static int scrambit[16];
static int bit[16];
```

```
static int bit_out[16];
static unsigned short lfsr = 0xffff; // 16 bit short for polynomial
int i, outbyte;

    if (inbyte == COMMA)   // if this is a comma
    {
        lfsr = 0xffff;     // reset the LFSR
        return (COMMA);    // and return the same data
    }


    if (inbyte == SKIP)    // don't advance or encode on skip
        return (SKIP);

    for (i=0; i<16;i++)    // convert LFSR to bit array for legibility
        bit[i] = (lfsr >> i) & 1;

    for (i=0; i<8; i++)    // convert byte to be scrambled for legibility
        scrambit[i] = (inbyte >> i) & 1;


    // apply the xor to the data
    if (! (inbyte & 0x100) &&    // if not a KCODE, scramble the data
        ! (TrainingSequence == TRUE))  // and if not in the middle of
    {                                  // a training sequence
        scrambit[0] ^= bit[15];
        scrambit[1] ^= bit[14];
        scrambit[2] ^= bit[13];
        scrambit[3] ^= bit[12];
        scrambit[4] ^= bit[11];
        scrambit[5] ^= bit[10];
        scrambit[6] ^= bit[9];
        scrambit[7] ^= bit[8];
    }

    // Now advance the LFSR 8 serial clocks
    bit_out[ 0] = bit[ 8];
    bit_out[ 1] = bit[ 9];
    bit_out[ 2] = bit[10];
    bit_out[ 3] = bit[11] ^ bit[ 8];
    bit_out[ 4] = bit[12] ^ bit[ 9] ^ bit[ 8];
    bit_out[ 5] = bit[13] ^ bit[10] ^ bit[ 9] ^ bit[ 8];
    bit_out[ 6] = bit[14] ^ bit[11] ^ bit[10] ^ bit[ 9];
```

```
    bit_out[ 7] = bit[15] ^ bit[12] ^ bit[11] ^ bit[10];
    bit_out[ 8] = bit[ 0] ^ bit[13] ^ bit[12] ^ bit[11];
    bit_out[ 9] = bit[ 1] ^ bit[14] ^ bit[13] ^ bit[12];
    bit_out[10] = bit[ 2] ^ bit[15] ^ bit[14] ^ bit[13];
    bit_out[11] = bit[ 3]           ^ bit[15] ^ bit[14];
    bit_out[12] = bit[ 4]                     ^ bit[15];
    bit_out[13] = bit[ 5];
    bit_out[14] = bit[ 6];
    bit_out[15] = bit[ 7];
    lfsr = 0;
    for (i=0; i <16; i++) // convert the LFSR back to an integer
        lfsr += (bit_out[i] << i);

    outbyte = 0;
    for (i= 0; i<8; i++) // convert data back to an integer
        outbyte += (scrambit[i] << i);

    return outbyte;
}

/*  NOTE THAT THE DESCRAMBLE ROUTINE IS IDENTICAL TO THE SCRAMBLE ROUTINE
    this routine implements the serial descrambling algorithm in parallel form
    this advances the lfsr 8 bits every time it is called
    this uses fewer than 25 xor gates to implement (with a static register)
    The XOR tree is the same as the scrambling routine
*/

int unscramble_byte(int inbyte)
{

    static int descrambit[8];
    static int bit[16];
    static int bit_out[16];
    static unsigned short lfsr = 0xffff;  // 16 bit short for polynomial
    int outbyte, i;
```

```
if (inbyte == COMMA)   // if this is a comma
{
   lfsr = 0xffff;      // reset the LFSR
   return (COMMA);     // and return the same data
}

if (inbyte == SKIP)   // don't advance or encode on skip
    return (SKIP);

for (i=0; i<16;i++)   // convert the LFSR to bit array for legibility
    bit[i] = (lfsr >> i) & 1;

for (i=0; i<8; i++)   // convert byte to be de-scrambled for legibility
    descrambit[i] = (inbyte >> i) & 1;

// apply the xor to the data
if (! (inbyte & 0x100) &&    // if not a KCODE, scramble the data
    ! (TrainingSequence == TRUE))  // and if not in the middle of
{                                  // a training sequence
    descrambit[0] ^= bit[15];
    descrambit[1] ^= bit[14];
    descrambit[2] ^= bit[13];
    descrambit[3] ^= bit[12];
    descrambit[4] ^= bit[11];
    descrambit[5] ^= bit[10];
    descrambit[6] ^= bit[9];
    descrambit[7] ^= bit[8];
}

// Now advance the LFSR 8 serial clocks
bit_out[ 0] = bit[ 8];
bit_out[ 1] = bit[ 9];
bit_out[ 2] = bit[10];
bit_out[ 3] = bit[11] ^ bit[ 8];
bit_out[ 4] = bit[12] ^ bit[ 9] ^ bit[ 8];
bit_out[ 5] = bit[13] ^ bit[10] ^ bit[ 9] ^ bit[ 8];
bit_out[ 6] = bit[14] ^ bit[11] ^ bit[10] ^ bit[ 9];
bit_out[ 7] = bit[15] ^ bit[12] ^ bit[11] ^ bit[10];
bit_out[ 8] = bit[ 0] ^ bit[13] ^ bit[12] ^ bit[11];
bit_out[ 9] = bit[ 1] ^ bit[14] ^ bit[13] ^ bit[12];
bit_out[10] = bit[ 2] ^ bit[15] ^ bit[14] ^ bit[13];
bit_out[11] = bit[ 3]           ^ bit[15] ^ bit[14];
```

```
    bit_out[12] = bit[ 4]                    ^ bit[15];
    bit_out[13] = bit[ 5];
    bit_out[14] = bit[ 6];
    bit_out[15] = bit[ 7];
    lfsr = 0;
    for (i=0; i <16; i++) // convert the LFSR back to an integer
        lfsr += (bit_out[i] << i);

    outbyte = 0;
    for (i= 0; i<8; i++) // convert data back to an integer
        outbyte += (descrambit[i] << i);

    return outbyte;
}
```

The initial 16-bit values of the LFSR for the first 128 LFSR advances following a reset are listed below:

	0, 8	**1, 9**	**2, A**	**3, B**	**4, C**	**5, D**	**6, E**	**7, F**
00	FFFF	E817	0328	284B	4DE8	E755	404F	4140
08	4E79	761E	1466	6574	7DBD	B6E5	FDA6	B165
10	7D09	02E5	E572	673D	34CF	CB54	4743	4DEF
18	E055	40E0	EE40	54BE	B334	2C7B	7D0C	07E5
20	E5AF	BA3D	248A	8DC4	D995	85A1	BD5D	4425
28	2BA4	A2A3	B8D2	CBF8	EB43	5763	6E7F	773E
30	345F	5B54	5853	5F18	14B7	B474	6CD4	DC4C
38	5C7C	70FC	F6F0	E6E6	F376	603B	3260	64C2
40	CB84	9743	5CBF	B3FC	E47B	6E04	0C3E	3F2C
48	29D7	D1D1	C069	7BC0	CB73	6043	4A60	6FFA
50	F207	1102	01A9	A939	2351	566B	6646	4FF6
58	F927	3081	85B0	AC5D	478C	82EF	F3F2	E43B
60	2E04	027E	7E72	79AE	A501	1A7D	7F2A	2197
68	9019	0610	1096	9590	8FCD	D0E7	F650	46E6
70	E8D6	C228	3AB2	B70A	129F	9CE2	FC3C	2B5C
78	5AA3	AF6A	70C7	CDF0	E3D5	C0AB	B9C0	D9C1

An 8-bit value of 0 repeatedly encoded with the LFSR after reset produces the following consecutive 8-bit values:

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	FF	17	C0	14	B2	E7	02	82	72	6E	28	A6	BE	6D	BF	8D
10	BE	40	A7	E6	2C	D3	E2	B2	07	02	77	2A	CD	34	BE	E0
20	A7	5D	24	B1	9B	A1	BD	22	D4	45	1D	D3	D7	EA	76	EE
30	2C	DA	1A	FA	28	2D	36	3B	3A	0E	6F	67	CF	06	4C	26
40	D3	E9	3A	CD	27	76	30	FC	94	8B	03	DE	D3	06	52	F6
50	4F	88	80	95	C4	6A	66	F2	9F	0C	A1	35	E2	41	CF	27
60	74	40	7E	9E	A5	58	FE	84	09	60	08	A9	F1	0B	6F	62
70	17	43	5C	ED	48	39	3F	D4	5A	F5	0E	B3	C7	03	9D	9B
80	8B	0D	8E	5C	33	98	77	AE	2D	AC	0B	3E	DA	0B	42	7A
90	7C	D1	CF	A8	1C	12	EE	41	C2	3F	38	7A	0D	69	F4	01
A0	DA	31	72	C5	A0	D7	93	0E	DC	AF	A4	55	E7	F0	72	16
B0	68	D5	38	84	DD	00	CD	18	9E	CA	30	59	4C	75	1B	77
C0	31	C5	ED	CF	91	64	6E	3D	FE	E8	29	04	CF	6C	FC	C4
D0	0B	5E	DA	62	BA	5B	AB	DF	59	B7	7D	37	5E	E3	1A	C6
E0	88	14	F5	4F	8B	C8	56	CB	D3	10	42	63	04	8A	B4	F7
F0	84	01	A0	01	83	49	67	EE	3E	2A	8B	A4	76	AF	14	D5
100	4F	AC	60	B6	79	D6	62	B7	43	E7	E5	2A	40	2C	6E	7A
110	56	61	63	20	6A	97	4A	38	05	E5	DD	68	0D	78	4C	53
120	8B	D6	86	57	B2	AA	1A	80	18	DC	BA	FC	03	A3	4B	30

At 2.5 GT/s, scrambling produces the power spectrum (in the 10-bit domain) shown in Figure C-1.

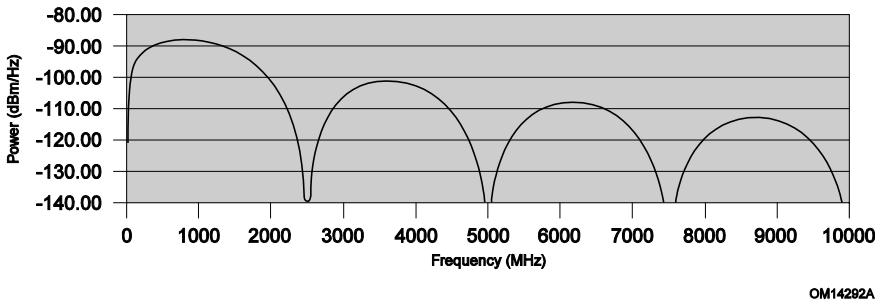


Figure C-1: Scrambling Spectrum at 2.5 GT/s for Data Value of 0

C.2. 128b/130b Data Scrambling Example

The following subroutines illustrate how calculate the next value of the 128b/130b scrambling LFSR and how to scramble (or descramble) an 8-bit value, both given the current value of the LFSR.

This is presented as one example only; there are many ways to obtain the proper output. This example demonstrates how to advance the LFSR eight times in one operation and how to XOR the data in one operation. Many other implementations are possible but they must all produce the same output as that shown here.

The following algorithm uses the "C" programming language conventions, where "<<" and ">>" represent the shift left and shift right operators, "^" is the exclusive or operator, and "|=" is the assignment by bitwise or operator.

```
#include <stdio.h>

// "Set Bit" - Sets bit number "bit" of "var" to the value "val".  Bit "bit" of "var" must start cleared.
#define SB(var,bit,val) var |= (val & 1) << bit

// "Get Bit" - Returns the value of bit number "bit" of "var".
#define GB(var,bit) ((var >> bit) & 1)

// Function to advance the LFSR value by 8 bits, given the current LFSR value
unsigned long int calc_next_lfsr(unsigned long int lfsr) {
  unsigned long int next_lfsr = 0;
  SB(next_lfsr,22, GB(lfsr,14) ^ GB(lfsr,16) ^ GB(lfsr,18) ^ GB(lfsr,20) ^ GB(lfsr,21) ^ GB(lfsr,22));
  SB(next_lfsr,21, GB(lfsr,13) ^ GB(lfsr,15) ^ GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,20) ^ GB(lfsr,21));
  SB(next_lfsr,20, GB(lfsr,12) ^ GB(lfsr,19) ^ GB(lfsr,21));
  SB(next_lfsr,19, GB(lfsr,11) ^ GB(lfsr,18) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(next_lfsr,18, GB(lfsr,10) ^ GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,21));
  SB(next_lfsr,17, GB(lfsr, 9) ^ GB(lfsr,16) ^ GB(lfsr,18) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(next_lfsr,16, GB(lfsr, 8) ^ GB(lfsr,15) ^ GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,21) ^ GB(lfsr,22));
  SB(next_lfsr,15, GB(lfsr, 7) ^ GB(lfsr,22));
  SB(next_lfsr,14, GB(lfsr, 6) ^ GB(lfsr,21));
  SB(next_lfsr,13, GB(lfsr, 5) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(next_lfsr,12, GB(lfsr, 4) ^ GB(lfsr,19) ^ GB(lfsr,21) ^ GB(lfsr,22));
  SB(next_lfsr,11, GB(lfsr, 3) ^ GB(lfsr,18) ^ GB(lfsr,20) ^ GB(lfsr,21) ^ GB(lfsr,22));
  SB(next_lfsr,10, GB(lfsr, 2) ^ GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,20) ^ GB(lfsr,21) ^ GB(lfsr,22));
  SB(next_lfsr, 9, GB(lfsr, 1) ^ GB(lfsr,16) ^ GB(lfsr,18) ^ GB(lfsr,19) ^ GB(lfsr,20) ^ GB(lfsr,21));
  SB(next_lfsr, 8, GB(lfsr, 0) ^ GB(lfsr,15) ^ GB(lfsr,17) ^ GB(lfsr,18) ^ GB(lfsr,19) ^ GB(lfsr,20));
  SB(next_lfsr, 7, GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,20) ^ GB(lfsr,21));
  SB(next_lfsr, 6, GB(lfsr,16) ^ GB(lfsr,18) ^ GB(lfsr,19) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(next_lfsr, 5, GB(lfsr,15) ^ GB(lfsr,17) ^ GB(lfsr,18) ^ GB(lfsr,19) ^ GB(lfsr,21) ^ GB(lfsr,22));
  SB(next_lfsr, 4, GB(lfsr,17));
```

```
  SB(next_lfsr, 3, GB(lfsr,16));
  SB(next_lfsr, 2, GB(lfsr,15) ^ GB(lfsr,22));
  SB(next_lfsr, 1, GB(lfsr,16) ^ GB(lfsr,18) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(next_lfsr, 0, GB(lfsr,15) ^ GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,21) ^ GB(lfsr,22));
  return(next_lfsr);
}

// Function to scramble a byte, given the current LFSR value
unsigned char scramble_data(unsigned long lfsr, unsigned char data_in) {
  unsigned char data_out = 0;
  SB(data_out, 7, GB(data_in,7) ^ GB(lfsr,15) ^ GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,21) ^ GB(lfsr,22));
  SB(data_out, 6, GB(data_in,6) ^ GB(lfsr,16) ^ GB(lfsr,18) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(data_out, 5, GB(data_in,5) ^ GB(lfsr,17) ^ GB(lfsr,19) ^ GB(lfsr,21));
  SB(data_out, 4, GB(data_in,4) ^ GB(lfsr,18) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(data_out, 3, GB(data_in,3) ^ GB(lfsr,19) ^ GB(lfsr,21));
  SB(data_out, 2, GB(data_in,2) ^ GB(lfsr,20) ^ GB(lfsr,22));
  SB(data_out, 1, GB(data_in,1) ^ GB(lfsr,21));
  SB(data_out, 0, GB(data_in,0) ^ GB(lfsr,22));
  return(data_out);
}


// Example of LFSR and scrambled data “0” sequence for Lane 0
main() {
  unsigned long lfsr = 0x1DBFBC; // Lane 0 reset LFSR value
  unsigned char unscrambled_data = 0x00;

  int i;
  printf("Iteration LFSR   Next LFSR Scrambled Data\n");
  printf("--------- ------ --------- --------------\n");
  for (i = 0; i < 128; i++) {
    unsigned char scrambled_data = scramble_data(lfsr,unscrambled_data);
    unsigned long next_lfsr = calc_next_lfsr(lfsr);
    printf("%3d       %06X    %06X             %02X\n",
           i, lfsr, next_lfsr, scrambled_data);
    lfsr = next_lfsr;
  }
}
```

The initial 23-bit values of the LFSR for the first 128 LFSR advances for an 8-bit quantity following a reset for Lane 0 are listed below (ordered left to right, top to bottom):

	0, 8	1, 9	2, A	3, B	4, C	5, D	6, E	7, F
00	1DBFBC	498C2E	1186E9	0FC5AD	7CB75D	3D8DA2	0ECC8F	379717
08	046BDF	21D462	423FCE	5177D6	1447EF	67CBA0	794F7A	6CA2AF
10	425060	3ED9D6	3DBF74	3C1A8F	7AE2E0	073E71	137D99	70B139
18	44F521	559720	1FBBA8	689D9F	376B02	597FFA	297C0E	7E450A
20	4BDE36	669B58	6BB530	78C3F9	0322C0	45C7FB	254F6A	323D89
28	07FDD2	71DFBC	68726B	799E27	1CFE8A	4AB864	42CB12	04AAF3
30	41F947	51F808	3A98CA	36A816	796895	4B4DAF	54037D	68E5C7
38	4F3302	60A51F	3AFCE3	528116	248131	1F65E6	17D2BA	34987E
40	6C0524	44DA45	7AF320	16FE71	5A5134	60B5F5	2A16E3	73AFF1
48	3D3ADA	18B5AA	4B9306	2BAB58	2D179E	5FDE68	46E1F8	644B91
50	3F78A4	7FCE1B	23CC59	7F017F	4DA97C	7EDF8B	705E13	0ADE04
58	6F1775	318CBE	70CC0C	39C021	0944ED	33F8AB	21DCBD	4AE0CE
60	1A6112	1B2F92	17ADD8	4BFA7E	42D358	1CE0F3	54C164	0BFDE2
68	0EF33F	082717	1200E1	4FCB73	39D53A	1C5FED	4ADE41	24EE12
70	7046E6	122B04	642E73	5A9AA4	0A24D0	34C250	362B24	5B5BB0
78	2833BF	73F640	648BDA	5E3281	490B97	373ECC	0CB1FA	6FE7A6

An 8-bit value of 0x00 repeatedly encoded with the LFSR after reset for Lane 0 produces the following consecutive 8-bit values (ordered left to right, top to bottom):

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	6C	BD	94	98	53	C6	D8	CE	50	6A	75	C1	04	4F	C3	07
10	75	26	C6	06	A3	B0	B4	AB	05	11	CC	57	4E	69	42	73
20	1D	0F	B7	03	E0	45	BA	5E	30	EB	D7	43	2C	5D	F5	D0
30	15	41	76	8E	C3	9D	D1	57	CD	FF	76	A1	7A	4C	64	2E
40	87	05	A3	24	89	FF	A2	4B	46	7C	1D	62	12	19	A5	2F
50	E6	B3	CA	33	ED	F3	2B	88	67	3E	AB	96	E8	9E	6A	5D
60	5C	1C	64	1D	F5	2C	51	C8	D8	A8	F4	4D	96	AC	5D	7A
70	2B	F4	2F	09	08	2E	0E	C9	02	4B	AF	D9	3D	4E	78	E7

D

D. Request Dependencies

The PCI Express Base Specification does not specify the rules governing the creation of resource dependencies between TLPs using different Traffic Classes. Dependencies between packets in different Traffic Classes can create potential deadlock if devices make different assumptions about what is allowed and what is not. Dependencies can be created when a packet is forwarded (transmitted verbatim) or translated (transmitted with modification) from an input Port to an output Port.

Resource dependencies are created when received packets are forwarded/translated on the same or a different Link. Due to the fact that the forwarding/translating device has finite buffer resources this behavior creates a dependency between the ability to receive a packet and the ability to transmit a packet (in potentially a different VC or sub-channel).

The following notation is used to create a framework to enumerate the possibilities:

X(m) -> Y(n)

This means:

a request in sub-channel X(TC=m) is forwarded/translated in sub-channel Y(TC=n).

n and m are between 0-7.

X and Y are either P (Posted Request), N (Non-Posted Request), or C (Completion).

The list of possible dependencies is:

P(m) -> P(n)

P(m) -> N(n)

P(m) -> C(n)

N(m) -> P(n)

N(m) -> N(n)

N(m) -> C(n)

C(m) -> P(n)

C(m) -> N(n)

C(m) -> C(n)

For a given system, each of these dependencies needs to be classified as legal or illegal for each of the following cases:

- Root Port forwarding to own Link.
- Root Port forwarding to different Root Port's Link.
- Endpoint or Bridge forwarding to own Link.

A Switch is not allowed to modify the TLPs that flow through it, but must ensure complete independence of resources assigned to separate VCs. System software must comprehend the system dependency rules when configuring TC/VC mapping throughout the system.

One possible legal mapping is:

	RC (Same Port)	RC (Different Port)	Endpoint
P(m) -> P(n)	m <= n	m <= n	m < n
P(m) -> N(n)	m < n	m < n	m < n
P(m) -> C(n)	illegal	illegal	illegal
N(m) -> P(n)	m < n	m < n	m < n
N(m) -> N(n)	m <= n	m <= n	m < n
N(m) -> C(n)	m = n	m = n	m = n
C(m) -> P(n)	illegal	illegal	illegal
C(m) -> N(n)	illegal	illegal	illegal
C(m) -> C(n)	m >= n	m >= n	m > n

Note that this discussion only deals with avoiding the deadlock caused by the creation of resource dependencies. It does not deal with the additional livelock issues (or lack of forward progress) caused by the system's Virtual Channel arbitration policies.

Some of these potential dependencies are illegal or unreachable:

- P(m) -> P(n), N(m) -> N(n)
 - m = n – This case is illegal and will lead to deadlock, except when a Request is being forwarded from one Port to another of a Switch or Root Complex.
 - m! = n – See discussion below.
- P(m) -> N(n)
 - m = n – This case is illegal and will lead to deadlock.
 - m! = n – See discussion below.
- N(m) -> P(n) – See discussion below.
- P(m) -> C(n) – This case is illegal and will lead to deadlock.

- N(m) -> C(n)
 - m = n – This case occurs during the normal servicing of a non-posted request by either a root complex or an endpoint.
 - m! = n – This case is unreachable and should never happen. Completions always use the same TC as the corresponding request.
- C(m) -> P(n), C(m) -> N(n) – These cases are unreachable and should never happen due to the fact that completion buffers must be preallocated.
- C(m) -> C(n)
 - m = n – This case is illegal and will lead to deadlock, except when a Completion is being forwarded from one Port to another of a Switch or Root Complex.
 - m! = n – This case will occur if N(n)->N(m) dependencies are allowed.

Other potential dependencies may be legal when comprehended as part of a specific usage model. For example, these cases:

P(m) ->P(n), N(m) -> N(n), P(m) -> N(n), N(m) -> P(n) where m! = n

might exist where a device services incoming requests by issuing modified versions of those requests using a different Traffic Class (for example, remapping the first requests address and generating the new request with the resulting address). In these cases, suitable rules must be applied to prevent circular dependencies that would lead to deadlock or livelock.

Examples of devices that may find the above mappings useful:

- Bridges to complex protocols that require state to be save/restored to/from host memory, i.e., PCI Express to Infiniband bridges.
- Messaging engines that must do address translation based upon page tables stored in host memory.
- UMA graphics devices that store their frame buffer in host memory.

E. ID-Based Ordering Usage

E.1. Introduction

ID-Based Ordering (IDO) is a mechanism that permits certain ordering restrictions to be relaxed as a means to improve performance. IDO permits certain TLPs to pass other TLPs in cases where otherwise such passing would be forbidden. The passing permitted by IDO is not required for proper operation (e.g., deadlock avoidance); it is only a means of improving performance.

For discussing IDO, it's useful to introduce the concept of a "TLP stream", which is a set of TLPs that all have the same originator.[110] For several important cases where TLP passing is normally forbidden, IDO permits such passing to occur if the TLPs belong to different TLP streams.

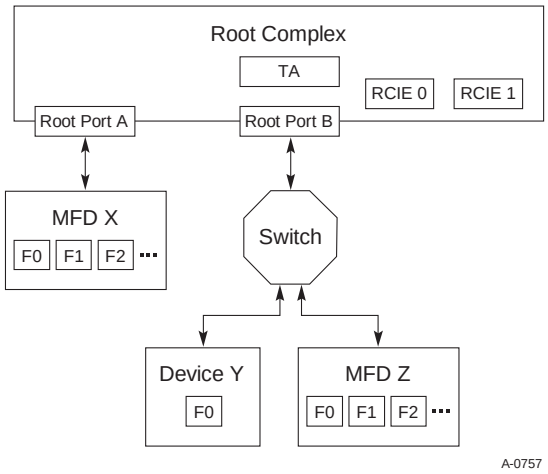


Figure E-1: Reference Topology for IDO Use

Figure E-1 shows a reference topology. The reference topology is not intended to discourage the use of IDO with other topologies, but rather to provide specific examples for discussion.

Devices X and Z are multi-Function devices (MFDs); device Y is a single-Function device. The RCIEs are Root Complex Integrated Endpoint Functions, and might or might not be part of the

[110] That is, the Requester IDs of Requests and the Completer IDs of Completions are all the same.

same Device. We will assume that one or more Functions are using the Translation Agent (TA) in the Root Complex (RC).

Referring to the ordering table and descriptions in Section 2.4.1, having the IDO bit set in a Posted Request, Non-Posted Request, or Completion TLP permits that TLP to pass a Posted Request TLP if the two TLPs belong to different TLP streams. In the following examples, DMAR and DMAW stand for Direct Memory Access Read and Write; PIOR and PIOW stand for Programmed I/O Read and Write.

E.2. Potential Benefits with IDO Use

Here are some example potential benefits that are envisioned with IDO use. Generally, IDO provides the most benefit when multiple TLP streams share a common Link and that Link becomes congested, either due to high utilization or due to temporary lack of Flow Control (FC) credit.

E.2.1. Benefits for MFD/RP Direct Connect

Here are some examples in the context of traffic between MFD X and the RC in Figure E-1.

- Posted Request passing another Posted Request: when a DMAW from F0 is stalled due to an TA miss, if IDO is set for a DMAW from F1, it is permitted within the RC for this DMAW to pass the stalled DMAW from F0.
- Non-Posted Request passing a Posted Request: when a DMAW from F0 is stalled due to an TA miss, if IDO is set for a DMAR Request from F1, it is permitted within the RC for this DMAR Request to pass the stalled DMAW from F0.
- Completion passing a Posted Request: when a DMAW from F0 is stalled due to an TA miss, if IDO is set for a PIOR Completion from F1, it is permitted within the RC for this PIOR Completion to pass the stalled DMAW from F0.

E.2.2. Benefits for Switched Environments

Here are some examples in the context of traffic within the Switch in Figure E-1.

- Non-Posted Request passing a Posted Request: when a DMAW from Device Y is stalled within the Switch due to a lack of FC credit from Root Port B, if IDO is set for a DMAR Request from MFD Z, it is permitted within the Switch for this DMAR Request to pass the stalled DMAW from Device Y. The same also holds for a DMAR Request from one Function in MFD Z passing a stalled DMAW from a different Function in MFD Z.
- Completion passing a Posted Request: when a DMAW from Device Y is stalled within the Switch due to a lack of FC credit from Root Port B, if IDO is set for a PIOR Completion from MFD Z, it is permitted within the Switch for this PIOR Completion to pass the stalled DMAW from Device Y. The same also holds for a PIOR Completion from one Function in MFD Z passing a stalled DMAW from a different Function in MFD Z.
- Posted Request passing another Posted Request: within a Switch, there is little or no envisioned benefit from having a DMAW from one TLP stream passing a DMAW from a different TLP

stream. However, it is not prohibited for Switches to implement such passing as permitted by IDO.

E.2.3. Benefits for Integrated Endpoints

Here are some examples for the Root Complex Integrated Endpoints (RCIEs) in Figure E-1. The benefits are basically the same as for the MFD/RP Direct Connect case.

- Posted Request passing another Posted Request: when a DMAW from RCIE 0 is stalled due to an TA miss, if IDO is set for a DMAW from RCIE 1, it is permitted for this DMAW to pass the stalled DMAW from RCIE 0.
- Non-Posted Request passing a Posted Request: when a DMAW from RCIE 0 is stalled due to an TA miss, if IDO is set for a DMAR Request from RCIE 1, it is permitted for this DMAR Request to pass the stalled DMAW from RCIE 0.
- Completion passing a Posted Request: when a DMAW from RCIE 0 is stalled due to an TA miss, if IDO is set for a PIOR Completion from RCIE 1, it is permitted for this PIOR Completion to pass the stalled DMAW from RCIE 0.

E.2.4. IDO Use in Conjunction with RO

IDO and RO[111] are orthogonal. Certain instances of passing; for example, a Posted Request passing another Posted Request, might be permitted by IDO, RO, or both at the same time. While IDO and RO have significant overlap for some cases, it is highly recommended that both be used whenever safely possible. RO permits certain TLP passing within the same TLP stream, which is never permitted by IDO. For traffic in different TLP streams, IDO permits control traffic to pass any other traffic, and generally it is not safe to Set RO with control traffic.

E.3. When to Use IDO

With Endpoint Functions[112], it is safe to Set IDO in all applicable TLPs originated by the Endpoint when the Endpoint is directly communicating with only one other entity, most commonly the RC. For the RC case, "directly communicating" specifically includes DMA traffic, PIO traffic, and interrupt traffic; communicating with RCIEs or communicating using P2P Root Port traffic constitutes communicating with multiple entities.

With a Root Port, there are no envisioned high-benefit use models where it is safe to Set IDO in all applicable TLPs that it originates. Use models where a Root Port Sets IDO in a subset of the applicable TLPs it originates are outside the scope of this specification.

[111] In this Appendix, "RO" is an abbreviation for the Relaxed Ordering Attribute field.

[112] Endpoint Functions include PCI Express Endpoints, Legacy PCI Express Endpoints, and Root Complex Integrated Endpoints.

E.4. When Not to Use IDO

E.4.1. When Not to Use IDO with Endpoints

With Endpoint Functions, it is not always safe to Set IDO in applicable TLPs it originates if the Endpoint directly communicates with multiple entities. It may be safe to Set IDO in some TLPs and not others, but such use models are outside the scope of this specification.

For example, in Figure E-1 if Device Y and MFD Z are communicating with P2P traffic and also communicating via host memory, it is not always safe for them to Set IDO in the TLPs they originate. As an example failure case, let's assume that Device Y does a DMAW (to host memory) followed by a P2P Write to MFD Z. Upon observing the P2P Write, let's assume that MFD Z then does a DMAW to the same location earlier targeted by the DMAW from Device Y. Normal ordering rules would guarantee that the DMAW from Device Y would be observed by host memory before the DMAW from MFD Z. However, if IDO is set in the DMAW from MFD Z, the RC would be permitted to have the second DMAW pass the first, causing a different end result in host memory contents.

Synchronization techniques like performing zero-length Reads might be used to avoid such communication failures when IDO is used, but specific use models are outside the scope of this specification.

E.4.2. When Not to Use IDO with Root Ports

With Root Ports, it is not always safe to Set IDO in applicable TLPs it originates if Endpoint Functions in the hierarchy do any P2P traffic. It may be safe to Set IDO in some TLPs and not others, but such use models are outside the scope of this specification.

As an example, in Figure E-1 if Device Y and MFD Z are communicating with P2P traffic and also communicating with host software, it is not always safe for Root Port B to Set IDO in the TLPs it originates. For example, let's assume that Device Y does a P2P Write to MFD Z followed by a DMAW (to host memory). Upon observing the DMAW, let's assume that the host does a PIOW to MFD Z. Normal ordering rules would guarantee that the P2P Write from Device Y would be observed by MFD Z before the PIOW from the host. However, if IDO is set in the PIOW from the host, the Switch would be permitted to have the PIOW pass the P2P Write, ultimately having the two Writes arrive at MFD Z out of order.

IMPLEMENTATION NOTE

Requester and Completer IDs for RC-Originated TLPs

With RC implementations where the Requester ID in a PIO Request does not match the Completer ID in a DMAR Completion, this enables another potential communication failure case if IDO is Set in the Completion. For this case, if a PIOW is followed by a DMAR Completion with IDO Set, a Switch below the Root Port could permit the DMAR Completion to pass the PIOW, violating the normal ordering rule that a non-RO Read Completion must not pass Posted Requests. The PIOW and DMAR Completion would appear to belong to different TLP streams, though logically they belong to the same TLP stream. Special caution is advised in setting IDO with TLPs originating from such RCs.

E.5. Software Control of IDO Use

E.5.1. Software Control of Endpoint IDO Use

By default, Endpoints are not enabled to Set IDO in any TLPs they originate.

IMPLEMENTATION NOTE

The "Simple" Policy for IDO Use

It is envisioned that Endpoints designed primarily to communicate directly with only one other entity (e.g., the RC) may find a "simple" policy for setting IDO to be adequate. Here's the envisioned "simple" policy. If the IDO Request Enable bit is Set, the Endpoint Sets IDO in all applicable Request TLPs that it originates. If the IDO Completion Enable bit is Set, the Endpoint Sets IDO in all Completion TLPs that it originates.

It is envisioned that a software driver associated with each Endpoint will determine when it is safe for the Endpoint to set IDO in applicable TLPs it originates. A driver should be able to determine if the Endpoint is communicating with multiple other entities, and should know the Endpoint's capabilities as far as setting IDO with all applicable TLPs when enabled, versus setting IDO selectively. If a driver determines that it is safe to enable the setting of IDO, the driver can set the IDO Request Enable and/or IDO Completion Enable bits either indirectly via OS services or directly, subject to OS policy.

If an Endpoint is designed for communication models where it is not safe to utilize the "simple" policy for IDO use, the Endpoint can implement more complex policies for determining when the Endpoint sets the IDO bit. Such implementations might utilize device-specific controls that are managed by the device driver. Such policies and device-specific control mechanisms are outside the scope of this specification.

E.5.2. Software Control of Root Port IDO Use

Since there are no envisioned high-benefit "simple" use models for Root Ports setting the IDO bit with TLPs they originate, and there are known communication failure cases if Root Ports set the IDO bit with all applicable TLPs they originate, it is anticipated that Root Ports will rarely be enabled to set IDO in TLPs they originate. Such use models and policies for Root Ports setting IDO are outside the scope of this specification.

F. Message Code Usage

Table F-1 contains a list of currently defined PCI Express Message Codes. Message codes are defined in this specification and in other specifications. This table will be updated as Messages are defined in other specifications but due to document release schedules, this table ~~may~~ might not contain recently defined Messages.

Comment [VS1]: ECN: Lightweight Notification (LN) Protocol

Table F-1: Message Code Usage

Message Code	Routing r[2:0]	Type	Description
0000 0000	011	Msg	Unlock, see Section 2.2.8.4
0000 0001	010	MsgD	Invalidate Request Message, see the *Address Translation Services Specification*
0000 0010	010	Msg	Invalidate Completion Message, see the *Address Translation Services Specification*
0000 0100	000	Msg	Page Request Message, see the *Address Translation Services Specification*
0000 0101	010	Msg	PRG Response Message, see the *Address Translation Services Specification*
0001 0000	100	Msg	Latency Tolerance Reporting (LTR) Message, see Section 2.2.8.8
0001 0010	100	Msg	Optimized Buffer Flush/Fill (OBFF) Message, see Section 2.2.8.9
0001 0100	100	Msg	PM_Active_State_Nak, see Section 2.2.8.2
0001 1000	000	Msg	PM_PME, see Section 2.2.8.2
0001 1001	011	Msg	PME_Turn_Off, see Section 2.2.8.2
0001 1011	101	Msg	PME_TO_Ack, see Section 2.2.8.2
0010 0000	100	Msg	Assert_INTA, see Section 2.2.8.1
0010 0001	100	Msg	Assert_INTB, see Section 2.2.8.1
0010 0010	100	Msg	Assert_INTC, see Section 2.2.8.1
0010 0011	100	Msg	Assert_INTD, see Section 2.2.8.1
0010 0100	100	Msg	Deassert_INTA, see Section 2.2.8.1
0010 0101	100	Msg	Deassert_INTB, see Section 2.2.8.1
0010 0110	100	Msg	Deassert_INTC, see Section 2.2.8.1
0010 0111	100	Msg	Deassert_INTD, see Section 2.2.8.1

Message Code	Routing r[2:0]	Type	Description
0011 0000	000	Msg	ERR_COR, see Section 2.2.8.3
0011 0001	000	Msg	ERR_NONFATAL, see Section 2.2.8.3
0011 0011	000	Msg	ERR_FATAL, see Section 2.2.8.3
0100 0000	100	Msg	Ignored Message, see Section 2.2.8.7
0100 0001	100	Msg	Ignored Message, see Section 2.2.8.7
0100 0011	100	Msg	Ignored Message, see Section 2.2.8.7
0100 0100	100	Msg	Ignored Message, see Section 2.2.8.7
0100 0101	100	Msg	Ignored Message, see Section 2.2.8.7
0100 0111	100	Msg	Ignored Message, see Section 2.2.8.7
0100 1000	100	Msg	Ignored Message, see Section 2.2.8.7
0101 0000	100	MsgD	Set_Slot_Power_Limit, see Section 2.2.8.5
0101 0010	100	Msg	PTM Request, see Section 2.2.8.10
0101 0011	100	Msg/MsgD	PTM Response/PTM ResponseD, see Section 2.2.8.10
0111 1110	000, 010, 011, or 100	Msg/MsgD	Vendor_Defined Type 0, see Section 2.2.8.6
0111 1111	000, 010, 011, or 100	Msg/MsgD	Vendor_Defined Type 1, see Section 2.2.8.6

Table F-2 contains a list of currently defined Subtype codes for PCI-SIG-Defined VDMs (see Section 2.2.8.6.1). Subtype codes are defined in this specification and in other specifications. This table will be updated as Subtype codes are defined in other specifications but due to document release schedules, this table might not contain recently defined Subtypes.

Table F-2: PCI-SIG-Defined VDM Subtype Usage

Subtype	Routing r[2:0]	Type	Description
0000 0000	010 or 011	MsgD	LN Message, see Section 2.2.8.6.1.1
0111 1111	100	Msg	Device Readiness Status, see Section 2.2.8.6.3
0111 1111	000	Msg	Function Readiness Status, see Section 2.2.8.6.4

Comment [VS2]: ECN: Precision Time Measurement

Comment [VS3]: ECN: Lightweight Notification (LN) Protocol

Comment [VS4]: ECN: Readiness Notification

Comment [VS5]: ECN: Readiness Notification

G

G. Protocol Multiplexing

The Protocol Multiplexing mechanism provides a standard mechanism to transport non-PCI Express protocols across a PCI Express Link. The mechanism supports the multiplexing of PMUX Packets and TLPs onto a single PCI Express Link.

An example system topology using Protocol Multiplexing is shown in Figure G-1. In this example, the Link may operate in a two modes:

- PCI Express Link. Protocol Multiplexing is disabled.
- PMUX Link. Protocol Multiplexing is enabled. Both TLPs and PMUX Packets are used in a coordinated fashion. PMUX Packets may be used to support additional protocols efficiently.

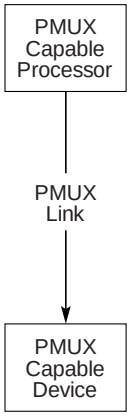


Figure G-1: Device and Processor Connected Using a PMUX Link

A PMUX Link is shown in Figure G-2. Arbitration and encapsulation occurs between the transmit queues and the Link. Demultiplexing and decapsulation occurs between the Link and the various receive queues. Packets are sent from transmit queues to the corresponding receive queues. Packets are identified as either PMUX Packets or TLPs.

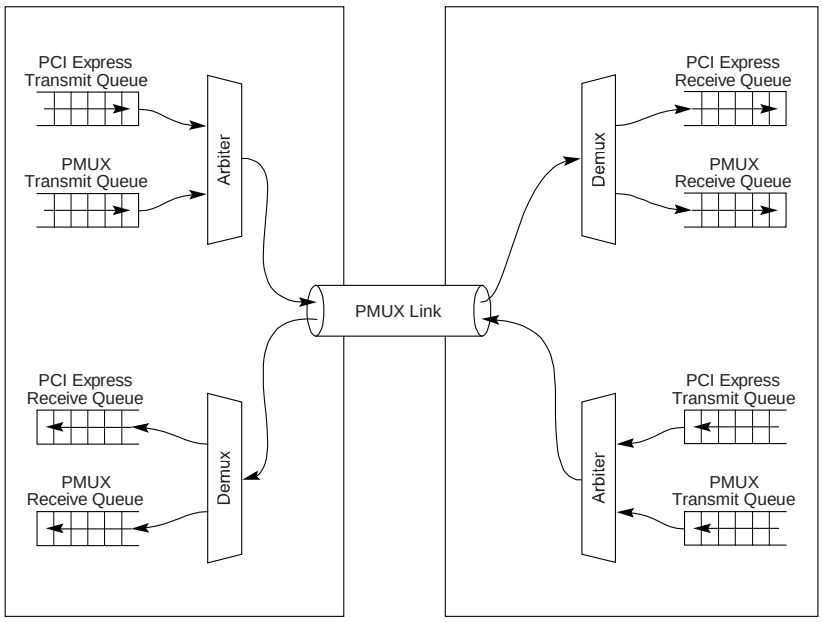


Figure G-2: PMUX Link

Important attributes of the Protocol Multiplexing mechanism are:

- Protocol Multiplexing support is optional normative.
- Protocol Multiplexing has no impact on PCI Express components that do not support it.
- Protocol Multiplexing has no impact on PCI Express TLPs and DLLPs, even when it is enabled.
- A Link may be used for both TLPs and PMUX Packets at the same time.
- Protocol Multiplexing does not consume or interfere with PCI Express resources (sequence numbers, credits, etc.). PMUX Packets use distinct resources associated with the specific multiplexed protocol.
- Protocol Multiplexing is disabled by default and is enabled by software. PMUX Packets must not be sent until enabled by software. PMUX Packets received at Ports that support Protocol Multiplexing are ignored until Protocol Multiplexing is enabled by software.
- Protocol Multiplexing is selectable on a per-Link basis. Protocol Multiplexing may be used on any collection of Links in a system.
- A PMUX Link may support up to four simultaneously active PMUX Channels. Software configures the protocol used on each PMUX Channel.

- PMUX Packets contain an LCRC. This is used to provide data resiliency in a similar fashion as PCI Express TLPs.
- PMUX Packets do not use the ACK/NAK mechanism of PCI Express. Multiplexed protocol specific acknowledgement mechanisms can be used to provide reliable delivery when needed.
- PMUX Packets do not contain a TLP Sequence Number. Instead, they contain a 12 bit PMUX Packet Metadata field that is available for multiplexed protocol specific use.
- PMUX Packet transmitters must contain some arbitration/QoS mechanism for scheduling sending of PMUX Packets, TLPs and DLLPs; however, the mechanism used is outside the scope of this specification.
- The Protocol Multiplexing mechanism does not define any addressing or routing mechanism for PMUX Packets.

PMUX Packets are similar to PCI Express TLPs. The PMUX Packet Flow Through is shown in Figure G-3. The PCI Express Packet Flow Through is shown in Figure 1-5. Changes from PCI Express Packet Flow Through are:

- PMUX Packets use a protocol specific PMUX Protocol Layer instead of the PCI Express Transaction Layer.
- PMUX Packets use a simplified Data Link Layer. The packet integrity portion of the Data Link Layer is mostly unchanged (LCRC computation uses a different seed value). The reliability and flow control aspects of the Data Link Layer are removed (the TLP Sequence Number field is repurposed as PMUX Packet Metadata).
- The Physical Layer is slightly modified to provide a mechanism to identify PMUX Packets.

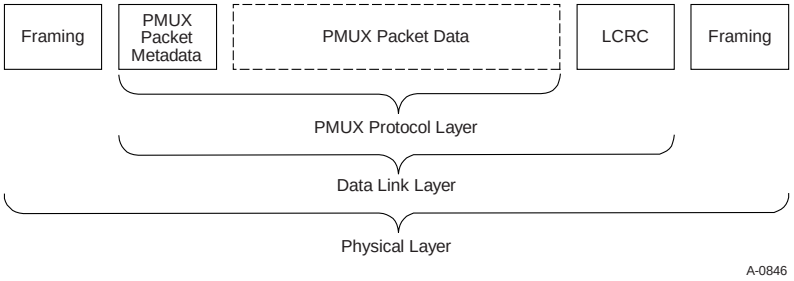


Figure G-3: PMUX Packet Flow Through the Layers

G.1. Protocol Multiplexing Interactions with PCI Express

Table G-1 and Table G-2 describe interactions between Protocol Multiplexing and PCI Express. Table G-1 describes how PCI Express attributes affect Protocol Multiplexing. Table G-2 describes how Protocol Multiplexing features affect PCI Express attributes.

Table G-1: PCI Express Attribute Impact on Protocol Multiplexing

PCI Express Attribute	Impact on Protocol Multiplexing
Link Speed	All PMUX Channels are disabled when the Current Link Speed corresponds to a speed that is not supported by Protocol Multiplexing (see Section G.5.4). A PMUX Channel may be disabled when the Current Link Speed corresponds to a speed that is not supported by the associated protocol.[113] Link speed can change either explicitly due to a change in the Target Link Speed field or automatically due to an autonomous Link speed change (see *PCI Express Base Specification* Section 6.11). The PMUX Protocol Layer is permitted to influence the mechanism used by a component to determine when it requests an autonomous Link speed change. In addition, setting Hardware Autonomous Speed Disable at each end of the Link will prevent certain autonomous Link speed changes (see Section 7.8.19). The PMUX Protocol Layer may be notified of the change.
Link Width	A PMUX Channel may be disabled when the Link Width corresponds to a width that is not supported by the associated protocol.[114] The PMUX Protocol Layer is permitted to influence the mechanism used by a component to determine when it requests a Link Width change. The PMUX Protocol Layer may be notified of the change.

[113] The mechanism software uses to determine what Link Speeds are supported by a protocol is outside the scope of this specification.

[114] The mechanism software uses to determine what Link Widths are supported by a protocol is outside the scope of this specification.

PCI Express Attribute	Impact on Protocol Multiplexing
FLR initiated	All PMUX Channels on a Link are disabled if an FLR is directed to the Upstream Port's Function 0. No PMUX Channels are affected if an FLR is directed to any other Function.
DL_Down	All PMUX Channels on a Link are disabled.
Hot Reset	All PMUX Channels on a Link are disabled.
PERST#	All PMUX Channels on a Link are disabled.
TLP Replay	No effect on Protocol Multiplexing.
DLLP Lost	No effect on Protocol Multiplexing.
TLP Prefix	No effect on Protocol Multiplexing.
Locked Transactions	No effect on Protocol Multiplexing (including no effect on any protocol specific forwarding of PMUX Packets).
AtomicOp Transactions	No effect on Protocol Multiplexing.
Multicast Transactions	No effect on Protocol Multiplexing.
Access Control Services (ACS)	No effect on Protocol Multiplexing.
Alternative Routing ID-Interpretation (ARI)	No effect on Protocol Multiplexing.
TLP Processing Hints (TPH)	No effect on Protocol Multiplexing.
Virtual Channels	No effect on Protocol Multiplexing. PCI Express Links remain capable of supporting a full complement of VCs.
Internal Error	Corrected or Uncorrectable Internal Errors in the PMUX Protocol Layer may be reported as PCI Express Internal Errors.
L0s Link Power State	Protocol Multiplexing tracks the Link state. The PMUX Protocol Layer may request the Link transition back to L0.
L1 Link Power State	Protocol Multiplexing tracks the Link state. The PMUX Protocol Layer may request the Link transition back to L0.
Disabled LTSSM State	Disabling a Link also disables all PMUX Channels on the Link.
Loopback LTSSM State	Entering Loopback state disables all PMUX Channels on the Link.
Recovery LTSSM State	No effect on Protocol Multiplexing. The PMUX Protocol Layer may be notified.
Receiver or Framing Error	The error is reported to the PMUX Protocol Layer to indicate that data might have been lost. This can be used to initiate protocol specific error recovery mechanisms. The Error is reported to software using PCI Express Mechanisms.

PCI Express Attribute	Impact on Protocol Multiplexing
Lane Reversal	No effect on Protocol Multiplexing. Support for Lane Reversal remains optional.
Polarity Inversion	No effect on Protocol Multiplexing.
Crosslink	No effect on Protocol Multiplexing. Support for Crosslink remains optional. If supported, the PMUX Protocol Layer may be notified of the outcome of the Crosslink Upstream / Downstream negotiation.
Lane assignment rules	Placement and frequency rules for STP Symbols and STP Tokens are not changed (see Section 4.2.1.2 and Section 4.2.2.3.2). These rules apply identically to PCI Express TLPs and PMUX Packets.
PCI Power Management Power State	All PMUX Channels on a Link are disabled if the Upstream Port's Function 0 is sent to non-D0 state.
Dynamic Power Allocation (DPA)	No effect on Protocol Multiplexing. The PMUX Protocol Layer is notified of the change and may participate in the power reduction. PCI Express power management includes any power used by the PMUX Protocol Layer.
PCI Power Management Power Consumed / Power Dissipated / Aux_Current	Power required by the PMUX Protocol Layer is included in the PCI structures.
Power Budgeting	Power required by the PMUX Protocol Layer is included in the PCI Express structures.
Slot Power Limit	Slot Power Limit includes power available to PMUX Protocol Layer.
ASPM L0s Entry Condition	The definition of Idle is extended to include: ❑ No pending PMUX Packets to transmit over the Link. ❑ For PMUX Channels that use protocol specific Flow Control, no credits are available to send PMUX Packets in that PMUX Channel.
ASPM L1 Entry Condition	A Link may not enter L1 if PMUX Packets are pending or scheduled to be transmitted.
ASPM L0s/L1 Exit Conditions	A Link may be directed to exit L0s or L1 if a component needs to transmit a PMUX Packet. Routing of PMUX Packets through routing elements is outside the scope of this specification; the associated L0s/L1 exit rules are also unspecified.
Bus Renumbering	No effect on Protocol Multiplexing.

PCI Express Attribute	Impact on Protocol Multiplexing
Hot Plug	No direct effect on Protocol Multiplexing. Note Hot Plug events indirectly affect Data Link State which, in turn affects Protocol Multiplexing.
TLP Sequence Number	No effect on Protocol Multiplexing. PMUX Packets do not consume TLP Sequence Numbers.
PCI Express Flow Control	No effect on Protocol Multiplexing. PMUX Packets do not consume PCI Express Flow Control credits. Flow Control Update DLLPs must be sent as required by PCI Express.
Error Reporting	No direct effect on Protocol Multiplexing. The PMUX Protocol Layer may be notified when an error is signaled or when an error message is received.
LCRC Errors in TLPs	No effect on Protocol Multiplexing.
Nullified TLPs	No effect on Protocol Multiplexing.
VC Arbitration	No effect on Protocol Multiplexing. Arbitration within PCI Express is unaffected by Protocol Multiplexing.
Port Arbitration	No effect on Protocol Multiplexing. Arbitration within PCI Express is unaffected by Protocol Multiplexing.
Electrical Idle Inference	PMUX Packets count as TLPs for the purpose of inferring Electrical Idle.
MR-IOV	Protocol Multiplexing may co-exist with MR-IOV. PMUX Packets are not part of any MR-IOV Virtual Hierarchy. Protocol Multiplexing is controlled using config space in the Management VH(s).

Table G-2: PMUX Attribute Impact on PCI Express

PMUX Attribute	Impact on PCI Express
PMUX Protocol Error	No effect on PCI Express.
LCRC Errors in PMUX Packets	No effect on PCI Express. PMUX Packets with LCRC errors are discarded without triggering PCI Express replay. This error is reported to the PMUX Protocol Layer and can be used to initiate protocol specific error recovery and/or error reporting mechanisms.
Link Unreliability	The PMUX Protocol Layer is permitted to influence the mechanism used by a component to determine if it requests an autonomous link speed change.
Nullified PMUX Packets	No effect on PCI Express. It is protocol specific whether PMUX Packets within a specific PMUX Channel may be nullified. If supported, PMUX Packets are nullified in the same manner as TLPs (e.g., inverting the LCRC and signaling nullification at the Physical Layer). Receiving a nullified PMUX Packet may be reported to the PMUX Protocol Layer.
Electrical Idle Inference	PMUX Packets count as TLPs for the purpose of inferring Electrical Idle.
PMUX Protocol Layer directs LTSSM to enter Recovery	Both PCI Express and the PMUX Protocol Layer are permitted to direct a transition from L0 to Recovery.
PMUX Channel Enabled / Disabled	No effect on PCI Express
PMUX Packet Receiver Buffer Overflow	No effect on PCI Express. This is a protocol problem within the PMUX Channel. The PMUX Transport Layer must continue to accept such packets dispose of them using protocol specific mechanisms.
Received PMUX Packet larger or smaller than supported by the associated protocol	No effect on PCI Express. These are protocol problems within the PMUX Channel. The PMUX Transport Layer must accept such packets and dispose of them using protocol specific mechanisms.
Received PMUX Packet that contains more than 125 DWORDs of PMUX Packet Data	No effect on PCI Express. This is an invalid PMUX Packet. The PMUX Transport Layer must accept such a packet and dispose of it. A protocol specific mechanism may be used to report the error. Note: This situation only exists for a packet encoded using 8b10b. The TLP Length field of a packet encoded using 128b/130b cannot contain values that cause this situation.

PMUX Attribute	Impact on PCI Express
PMUX Packet Received on disabled PMUX Channel	No effect on PCI Express. No effect on any other PMUX Channel. Receivers must silently ignore such packets regardless of packet length and regardless of whether or not the packet is nullified. PMUX Packets arriving on a disabled PMUX Channel may occur normally when software is in the process of initializing Protocol Multiplexing.
PMUX Packet Received at component that does not support Protocol Multiplexing	Software should not enable PMUX Packets unless both ends of a Link support Protocol Multiplexing. In the 128b/130b encoding, receiving a PMUX Packet by a component that does not support Protocol Multiplexing is a Framing Error (see Section 4.2.2.3.1). In the 8b10b encoding, the PMUX Packet LCRC is computed differently that the TLP LCRC. Receivers that do not support Protocol Multiplexing will interpret PMUX Packets as TLPs with LCRC errors and will not process them.
Large PMUX Packets when PCI Express Max_Payload_Size is small	Under certain conditions, it is possible for a large PMUX Packet to trigger a premature PCI Express replay. For example, this can occur when the time needed to transmit a PMUX Packet is larger than the REPLAY_TIMER (see Section 3.5.2.1). To avoid this issue, implementations are permitted to not advance (hold) their REPLAY_TIMER during the reception of PMUX Packets. Note: The PCI Express REPLAY_TIMER mechanism has adequate headroom for most cases. This issue exists when (1) Max_Payload_Size is 000b, (2) PMUX Packets are larger than about 80 DWORDs, and (3) the REPLAY_TIMER is at the low end of the -0%/+100% tolerance.

G.2. PMUX Packets

A PMUX Packet contains the information shown in Figure G-4.

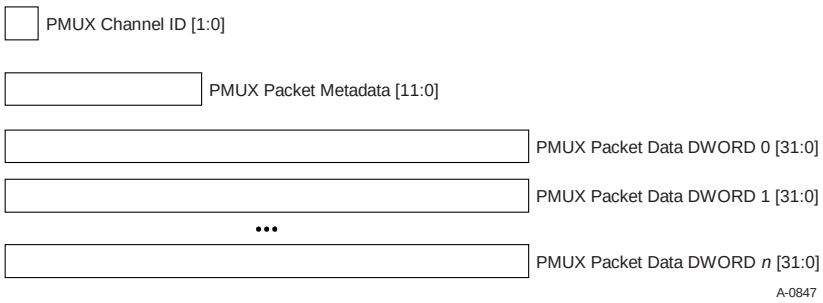


Figure G-4: PMUX Packet

PMUX Channel ID is a 2 bit field that identifies which protocol is associated with a PMUX Packet. PMUX Channel ID values are between 0 and 3 (inclusive).

PMUX Packet Metadata is a 12 bit field that provides information about the PMUX Packet. Definition of this field is protocol specific and is outside the scope of this specification.

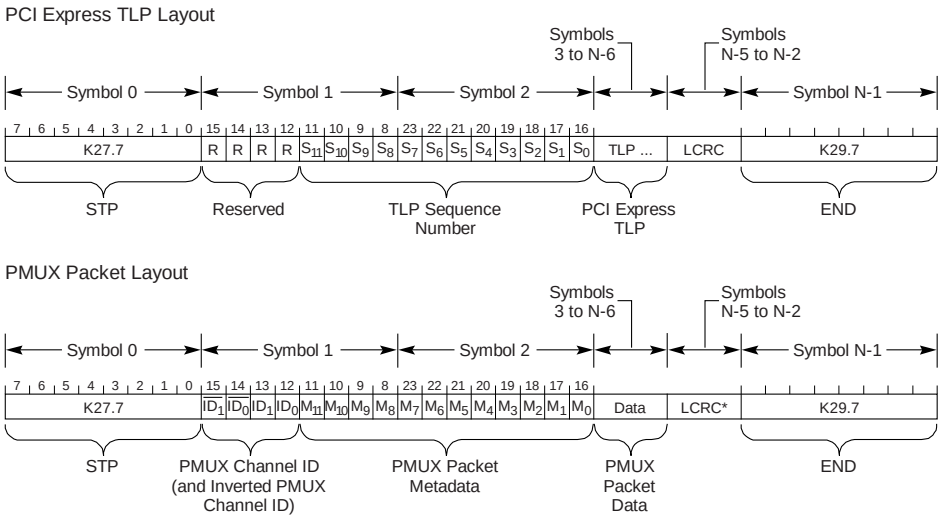
A PMUX Packet consists of between 0 and 125 DWORDs of PMUX Packet Data. Layout and usage of these DWORDs is protocol specific and is outside the scope of this specification. A PMUX Packet need not have any PMUX Packet Data and may consist only of PMUX Channel ID and PMUX Packet Metadata.

G.3. PMUX Packet Layout

There are two layouts defined for PMUX Packets. One layout is used for 2.5 and 5.0 GT/s data rates and another layout is used for 8.0 GT/s and higher data rates. These layouts are discussed in the following sections.

G.3.1. PMUX Packet Layout for 8b10b Encoding

Figure G-5 and Table G-3 show the layout of PMUX Packets when using 8b10b encoding. For reference, the 8b10b encoding of a TLP is also shown (see Section 4.2.1.2 for the official definition). In Table G-3, items shown in *italics* are identical in PMUX Packets and TLPs.



* LCRC for PMUX Packets uses a different starting seed than that used for TLPs. This avoids aliasing problems and potential errors if software misconfigures a link so that PMUX Packets are seen by existing silicon.

A-0848

Figure G-5: TLP and PMUX Packet Framing (8b10b Encoding)

Table G-3: PMUX Packet Layout (8b10b Encoding)

Symbol	Field	Bit Position(s)	PMUX Packet Usage	TLP Usage
0	*Start TLP Indicator*	*7:0*	*K27.7*	
1	Inverted PMUX Channel ID [1:0]	7:6	Inverted (1s complement) of Symbol 1 bits 6:4	Reserved
	PMUX Channel ID[1:0]	5:4	PMUX Channel ID	Reserved
	PMUX Packet Metadata[11:8]	3:0	PMUX Packet Metadata[11:8]	TLP Sequence Number[11:8]
2	PMUX Packet Metadata[7:0]	7:0	PMUX Packet Metadata[7:0]	TLP Sequence Number[7:0]
3 to N-6	Packet	7:0	PMUX Packet	TLP
N-5 to N-2	LCRC	7:0	PMUX LCRC	PCI Express LCRC
N-1	*END*	*7:0*	*K29.7*	

For PMUX Packets, symbols 1 and 2 contain PMUX Packet Metadata in the same bit positions that TLPs use for TLP Sequence Number.

The PMUX LCRC algorithm is identical to the TLP LCRC algorithm as described in Section 3.5.2 with the following modifications:

- The seed value is FB3E E248h (TLP LCRC uses FFFF FFFFh).
- The PMUX Channel ID field in Symbol 1 bits 7:4 is included in the PMUX LCRC in the same manner as the 4 reserved bits in the TLP LCRC.
- The PMUX Packet Metadata field is included in the PMUX LCRC in the same manner as the TLP Sequence Number field is included in the TLP LCRC.



IMPLEMENTATION NOTE

PMUX Packets at Receivers that do not Support Protocol Multiplexing

The bits used for PMUX Channel ID are reserved unless Protocol Multiplexing is supported. As such, Receivers that do not support Protocol Multiplexing must ignore the PMUX Channel ID bits. If software misconfigures Protocol Multiplexing, a component that does not support Protocol Multiplexing could receive a PMUX Packet. To prevent that component from misinterpreting such a PMUX Packet as a valid TLP, the LCRC computation is changed for PMUX Packets. The result is that a valid PMUX Packet will never be misinterpreted as a valid TLP. These LCRC "errors" may trigger PCI Express replay and may result in REPLAY_NUM Rollover correctable errors being reported.

IMPLEMENTATION NOTE

PMUX Packet LCRC

The PMUX Channel ID field is covered by the LCRC. As such, when using 8b10b encoding, receivers must wait until the LCRC is checked to make firm decisions based on the PMUX Channel ID value. The Inverted PMUX Channel ID can be compared against the PMUX Channel ID to make tentative decisions.

Note: The value of the LCRC associated with a given PMUX Packet is independent of the encoding used to transmit the packet.

G.3.2. PMUX Packet Layout at 128b/130b Encoding

Figure G-6 and Table G-4 show the layout of PMUX Packets when using 128b/130b encoding. For reference, the 128b/130b encoding of a TLP is also shown (see Section 4.2.2.2 for the official definition). In Table G-4, items shown in *italics* are identical in PMUX Packets and TLPs.

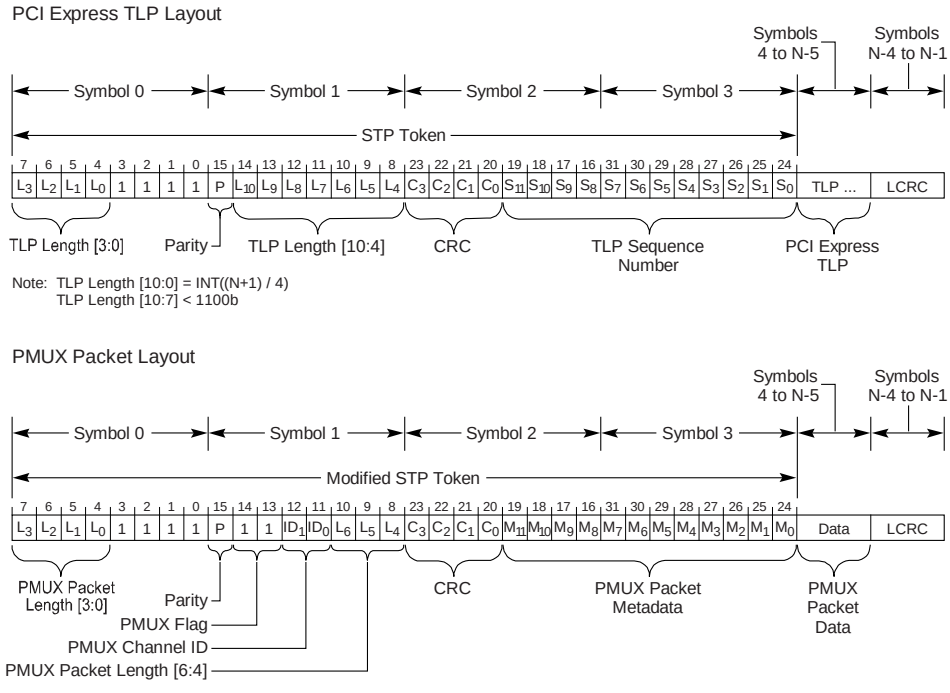


Figure G-6: TLP and PMUX Packet Framing (128b/130b Encoding)

Table G-4: PMUX Packet Layout (128b/130b Encoding)

Symbol	Field	Bit Position(s)	PMUX Packet Usage	TLP Usage
0	*Start TLP Indicator*	*3:0*	*Value of 1111b*	
	PMUX Packet Length[3:0]	7:4	Bits [3:0] of the PMUX Packet Length. Bit 0 is the least significant PMUX Packet Length bit.	Bits [3:0] of the TLP Length field. Bit 0 is the least significant TLP Length bit.
1	*Frame Parity (P)*	*7*	*Even parity of Symbol 0 bits [7:4], Symbol 1 bits [6:0] and Symbol 2 bits [7:4]*	
	PMUX Packet Indicator	6:5	Value of 11b	Bits [10:4] of the TLP Length field. Bit 10 is the most significant TLP Length bit.
	PMUX Channel ID[1:0]	4:3	PMUX Channel ID	
	PMUX Packet Length[6:4]	2:0	Bits [6:4] of PMUX Packet Length. Bit 6 is the most significant PMUX Packet Length bit.	
2	PMUX Packet Metadata[11:8]	3:0	PMUX Packet Metadata[11:8]	TLP Sequence Number[11:8]
	Frame CRC (C[3:0])	*7:4*	*CRC of Symbol 0, bits [7:4] and Symbol 1 bits [6:0]*	
3	PMUX Packet Metadata[7:0]	7:0	PMUX Packet Metadata[7:0]	TLP Sequence Number[7:0]
4 to N-5	Packet	7:0	PMUX Packet	TLP
N-4 to N-1	*LCRC*	*7:0*	*LCRC*	

Table G-5 describes the encodings of Symbol 1 bits [6:3] in more detail. If these bits contain a value less than 1001b, the packet is a TLP and is processed as described in Section 4.2.2.[115] If these bits contain 1001b, 1010b, or 1011b, the encoding is reserved for future standardization and is processed as described in Section 4.2.2.3.3. If these bits contain a value greater than or equal to 1100b, the packet is a PMUX Packet is defined as specified in this appendix.[116]

[115] The value 1001b supports a maximum TLP Length [10:0] value of 1151 DWORDs (decimal).This will accommodate a TLP consisting of 4096 bytes of payload, 16 bytes of TLP Header, 4 bytes of TLP digest, and 480 bytes of TLP Prefix.

[116] The value 1100b was chosen to simplify distinguishing PMUX Packets from TLPs and from the reserved encodings.

Table G-5: Symbol 1 Bits [6:3]

Symbol 1 bits [6:3]	Meaning
0xxxb or 1000b	Packet is a TLP. Bits [6:3] are TLP Length [10:7].
1001b, 1010b, or 1011b	Encoding reserved for future standardization. Receivers detecting these encodings shall process them as described in Section 4.2.2.3.3.
1100b	Packet is a PMUX Packet. PMUX Channel ID is 0.
1101b	Packet is a PMUX Packet. PMUX Channel ID is 1.
1110b	Packet is a PMUX Packet. PMUX Channel ID is 2.
1111b	Packet is a PMUX Packet. PMUX Channel ID is 3.

For PMUX Packets, the packet length in DWORDs is contained in PMUX Packet Length [6:0]. Other than being a smaller field, PMUX Packet Length is interpreted in the same manner as TLP Length. Specifically, PMUX Packet Length also includes the framing and PMUX LCRC DWORDs (see Section 4.2.2.2).

For PMUX Packets, symbols 2 and 3 contain PMUX Packet Metadata in the same bit positions that TLPs use for TLP Sequence Number.

The PMUX LCRC algorithm is identical to the TLP LCRC algorithm as described in Section 3.5.2 with the following modifications:

- The seed value is FB3E E248h (TLP LCRC uses FFFF FFFFh).
- The PMUX Channel ID field in Symbol 1 bits 4:3 is used to compute a 4 bit value that is included in the PMUX LCRC in the same manner as the 4 reserved bits in the TLP LCRC. This 4 bit value contains the value that would be used, by the 8b10b encoding, for Symbol 1 bits 7:4. Specifically, the lower 2 bits of this 4 bit value contain the PMUX Channel ID and the upper 2 bits contain the inverse (1s complement) of the PMUX Channel ID.
- The PMUX Packet Metadata field is included in the PMUX LCRC in the same manner as the TLP Sequence Number field is included in the TLP LCRC.

The Frame CRC and Frame Parity fields are computed as shown below. This is the same algorithm computed over the same bit positions as defined in Section 4.2.2.2.

C[0] = 1b ^ PMUX_Channel_ID[0] ^ L[6] ^ L[4] ^ L[2] ^ L[1] ^ L[0]

C[1] = 1b ^ 1b ^ PMUX_Channel_ID[0] ^ L[5] ^ L[4] ^ L[3] ^ L[2]

C[2] = 1b ^ PMUX_Channel_ID[1] ^ L[6] ^ L[4] ^ L[3] ^ L[2] ^ L[1]

C[3] = PMUX_Channel_ID[1] ^ PMUX_Channel_ID[0] ^ L[5] ^ L[3] ^ L[2] ^ L[1] ^ L[0]

P = 1b ^ 1b ^ PMUX_Channel_ID[1] ^ PMUX_Channel_ID[0] ^ L[6] ^ L[5] ^ L[4] ^ L[3] ^ L[2] ^ L[1] ^ L[0] ^ C[3] ^ C[2] ^ C[1] ^ C[0]d



IMPLEMENTATION NOTE

PMUX Channel ID and Frame CRC

When using 128b/130b encoding, the PMUX Channel ID field is covered by the Frame CRC and Frame Parity fields. As such, receivers may make decisions based on the PMUX Channel ID value as soon as the Frame CRC and Frame Parity is checked and need not wait until the PMUX LCRC is checked.

Note: The PMUX Channel ID is also covered by the LCRC. The value of the LCRC associated with a given PMUX Packet is independent of the encoding used to transmit the packet.

G.4. PMUX Control

Protocol Multiplexing is disabled by default. Each PMUX Channel must be explicitly enabled by software at each end of the associated Link. Protocol Multiplexing is disabled whenever the link drops (Data Link Layer indicates DL_Down).

A component that supports Protocol Multiplexing indicates such by the presence of the PMUX Extended Capability.

The following rules apply to components that support Protocol Multiplexing:

- PMUX Packets received in a PMUX Channel that is not enabled are silently ignored.
- PMUX Packets may not be transmitted unless the associated PMUX Channel is enabled. A PMUX Channel may also require additional, protocol specific, initialization mechanisms before PMUX Packets may be transmitted.

G.5. PMUX Extended Capability

Figure G-7 shows the PMUX Extended Capability structure. The presence of this capability indicates that the Port supports the optional Protocol Multiplexing mechanism. This capability is optional and may be present in any Downstream Port and in Function 0 of any Upstream Port. It must not be present in non-zero Functions of Upstream Ports or in RCRBs.

The length of the PMUX Extended Cpability is determined by the PMUX Protocol Array Size field (see Section G.5.2).

This capability contains a list of the protocols supported by the Link (the PMUX Protocol Array). It also contains the mechanism software uses to enable and configure PMUX Channels. This capability must be present in both the Upstream and Downstream Ports of a Link in order for Protocol Multiplexing to be successfully enabled.

Software may enable the Upstream and Downstream Ports of a Link in either order. Software may enable multiple PMUX Channels using a single write to the PMUX Control Register.

Behavior is undefined if software enables Protocol Multiplexing in one Port and the other Port of the Link does not support Protocol Multiplexing. Behavior is also undefined if software configures a PMUX Channel inconsistently (the same PMUX Channel in the Ports on each end of a Link configured with incompatible protocols).

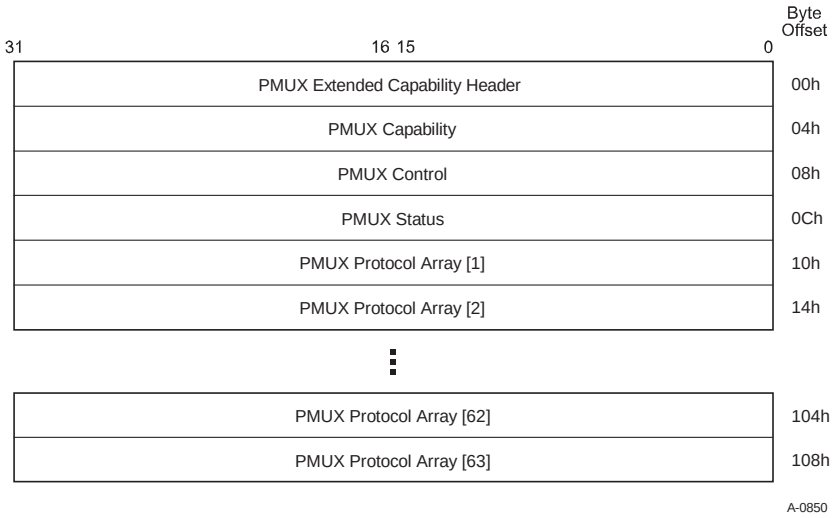


Figure G-7: PMUX Extended Capability

G.5.1. PCI Express Extended Header (Offset 00h)

Figure G-8 details the allocation of fields in the PMUX Extended Capability header; Table G-6 provides the respective bit definitions.

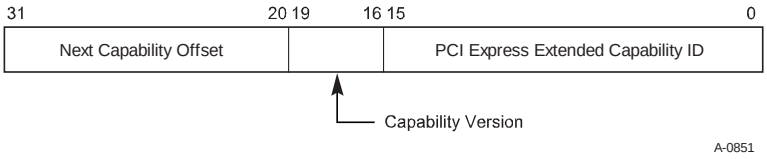


Figure G-8: PMUX Extended Capability Header

Table G-6: PMUX Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. The Extended Capability ID for the PMUX Extended Capability is 001Ah.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.	RO

Bit Location	Register Description	Attributes
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities. This offset is relative to the beginning of PCI compatible Configuration Space and thus must always be either 000h (for terminating the list of Capabilities) or greater than 0FFh.	RO

G.5.2. PMUX Capability Register (Offset 04h)

Figure G-9 details the allocation of fields in the PMUX Capability register. Table G-7 provides the respective bit definitions.

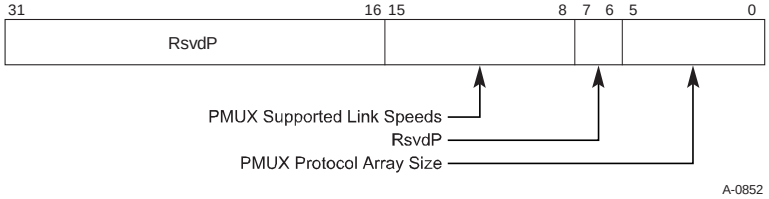


Figure G-9: PMUX Capability Register

Table G-7: PMUX Capability Register

Bit Location	Register Description	Attributes
5:0	**PMUX Protocol Array Size** – Indicates the size of this Function's PMUX Protocol Array. This field may be 0 to indicate that even though no protocols are supported, the Port will ignore all received PMUX Packets.	RO
7:6	**Reserved**	RsvdP

Bit Location	Register Description	Attributes
15:8	**PMUX Supported Link Speeds** – This field indicates the Link speed(s) where Protocol Multiplexing is supported. Each bit corresponds to a Link speed. If a bit is Set, Protocol Multiplexing is supported at that Link speed. If a bit is Clear, Protocol Multiplexing is not supported at that Link speed. Bit definitions are: Bit 8 2.5 GT/s Bit 9 5.0 GT/s Bit 10 8.0 GT/s Bits 15:11 RsvdP At least one Link speed must be supported (i.e., the field must be non-zero). A Port may support any combination of Link speeds. For example, this field could contain the value 0000 0100b indicating that Protocol Multiplexing is only supported at 8.0 GT/s. This field must not indicate support for Link speeds that are not supported by the Link (see Section 7.8.18). Note that this field indicates the Link speeds supported by Protocol Multiplexing for the Link. The Link speeds that a particular protocol supports and the mechanism used to report that information are protocol specific.	RO / RsvdP
31:16	**Reserved**	RsvdP

G.5.3. PMUX Control Register (Offset 08h)

Figure G-10 details the allocation of fields in the PMUX Control register. Table G-8 provides the respective bit definitions.

Channel *n* is enabled and available for use by the PMUX Protocol Layer when all of the following are true:

- The Channel *n* Assignment field is non-zero.
- The Channel *n* Assignment field is less than or equal to PMUX Prototocol Array Size.
- The Channel *n* Assignment field indicates an implemented entry in the PMUX Protocol Array (see Section G.5.5).
- All of the PMUX Channel *n* Disabled bits are Clear (see Section G.5.4).

Otherwise, Channel *n* is disabled.

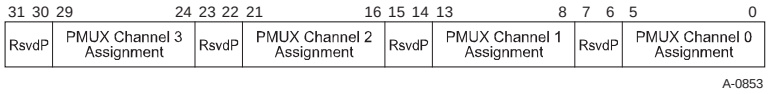


Figure G-10: PMUX Control Register

Table G-8: PMUX Control Register

Bit Location	Register Description	Attributes
5:0	**PMUX Channel 0 Assignment** – This field indicates the protocol assigned to PMUX Channel 0. If the field is 0h, no protocol is assigned. If the field is non-zero, it is the index in the PMUX Protocol Array of the protocol assigned to PMUX Channel 0. If PMUX Protocol Array Size is less than 63 (see Section G.5.2), unused upper bits of this field may be hardwired to 0b. If PMUX Protocol Array Size is 0, this entire field may be hardwired to 0. This field defaults to 0h.	RW
7:6	**Reserved**	RsvdP
13:8	**PMUX Channel 1 Assignment** – This field indicates the protocol assigned to PMUX Channel 1. If the field is 0h, no protocol is assigned. If the field is non-zero, it is the index in the PMUX Protocol Array of the protocol assigned to PMUX Channel 1. If PMUX Protocol Array Size is less than 63 (see Section G.5.2), unused upper bits of this field may be hardwired to 0b. If PMUX Protocol Array Size is 0, this entire field may be hardwired to 0. This field defaults to 0h.	RW
15:14	**Reserved**	RsvdP
21:16	**PMUX Channel 2 Assignment** – This field indicates the protocol assigned to PMUX Channel 2. If the field is 0h, no protocol is assigned. If the field is non-zero, it is the index in the PMUX Protocol Array of the protocol assigned to PMUX Channel 2. If PMUX Protocol Array Size is less than 63 (see Section G.5.2), unused upper bits of this field may be hardwired to 0b. If PMUX Protocol Array Size is 0, this entire field may be hardwired to 0. This field defaults to 0h.	RW
23:22	**Reserved**	RsvdP

Bit Location	Register Description	Attributes
29:24	**PMUX Channel 3 Assignment** – This field indicates the protocol assigned to PMUX Channel 3. If the field is 0h, no protocol is assigned. If the field is non-zero, it is the index in the PMUX Protocol Array of the protocol assigned to PMUX Channel 3. If PMUX Protocol Array Size is less than 63 (see Section G.5.2), unused upper bits of this field may be hardwired to 0b. If PMUX Protocol Array Size is 0, this entire field may be hardwired to 0. This field defaults to 0h.	RW
31:30	**Reserved**	RsvdP

G.5.4. PMUX Status Register (Offset 0Ch)

Figure G-11 details the allocation of fields in the PMUX Status register. Table G-9 provides the respective bit definitions.

Each channel has a set of Disabled bits. When Channel *n* Assignment field is non-zero, the Channel *n* Disabled bits reflect the error status of the channel. The following Disabled bits are defined:

- PMUX Channel *n* Disabled: Link Speed
- PMUX Channel *n* Disabled: Link Width
- PMUX Channel *n* Disabled: Protocol Specific

When there are multiple reasons for disabling a channel, an implementation may choose which reason(s) to report. For example, if a protocol needs bandwidth equivalent to x1 8.0 GT/s, when there is inadequate bandwidth (e.g., the Link is operating at x1 5.0 GT/s, x1 2.5 GT/s, or x2 2.5 GT/s), it could disable the PMUX Channel by indicating any or all of Disabled: Link Width, Disabled: Link Speed, or Disabled: Protocol Specific.

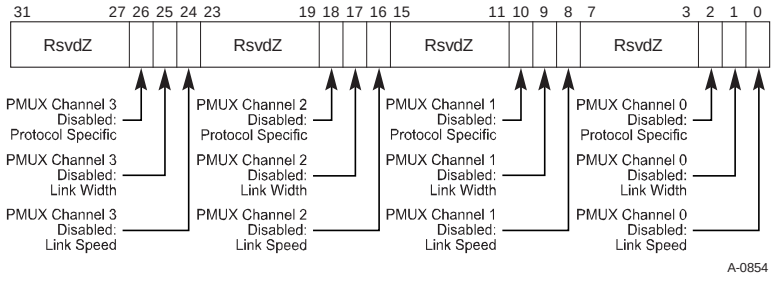


Figure G-11: PMUX Status Register

Table G-9: PMUX Status Register

Bit Location	Register Description	Attributes
0	**PMUX Channel 0 Disabled: Link Speed** – If Set, Channel 0 is disabled because the Current Link Speed (Section 7.8.8) is not supported by Protocol Multiplexing or by the protocol assigned to Channel 0. This bit is 0 when no protocol is assigned to Channel 0 (i.e., Channel 0 Control field is 0h).	RO
1	**PMUX Channel 0 Disabled: Link Width** – If Set, Channel 0 is disabled because the current Link Width is not supported by the protocol assigned to Channel 0. This bit is 0 when no protocol is assigned to Channel 0 (i.e., Channel 0 Assignment field is 0h).	RO
2	**PMUX Channel 0 Disabled: Protocol Specific** – If Set, Channel 0 is disabled for protocol specific reasons. This bit is 0 when no protocol is assigned to Channel 0 (i.e., Channel 0 Assignment field is 0h).	RO
7:3	**Reserved**	RsvdZ
8	**PMUX Channel 1 Disabled: Link Speed** – If Set, Channel 1 is disabled because the Current Link Speed (Section 7.8.8) is not supported by Protocol Multiplexing or by the protocol assigned to Channel 1. This bit is 0 when no protocol is assigned to Channel 1 (i.e., Channel 1 Assignment field is 0h).	RO
9	**PMUX Channel 1 Disabled: Link Width** – If Set, Channel 1 is disabled because the current Link Width is not supported by the protocol assigned to Channel 1. This bit is 0 when no protocol is assigned to Channel 1 (i.e. Channel 1 Assignment field is 0h).	RO
10	**PMUX Channel 1 Disabled: Protocol Specific** – If Set, Channel 1 is disabled for protocol specific reasons. This bit is 0 when no protocol is assigned to Channel 1 (i.e., Channel 1 Assignment field is 0h).	RO
15:11	**Reserved**	RsvdZ
16	**PMUX Channel 2 Disabled: Link Speed** – If Set, Channel 2 is disabled because the Current Link Speed (Section 7.8.8) is not supported by Protocol Multiplexing or by the assigned protocol. This bit is 0 when no protocol is assigned to Channel 2 (i.e., Channel 2 Assignment field is 0h).	RO

Bit Location	Register Description	Attributes
17	**PMUX Channel 2 Disabled: Link Width** – If Set, Channel 2 is disabled because the current Link Width is not supported by the assigned protocol. This bit is 0 when no protocol is assigned to Channel 2 (i.e., Channel 2 Assignment field is 0h).	RO
18	**PMUX Channel 2 Disabled: Protocol Specific** – If Set, Channel 2 is disabled for protocol specific reasons. This bit is 0 when no protocol is assigned to Channel 2 (i.e., Channel 2 Assignment field is 0h).	RO
23:19	**Reserved**	RsvdZ
24	**PMUX Channel 3 Disabled: Link Speed** – If Set, Channel 3 is disabled because the Current Link Speed (Section 7.8.8) is not supported by Protocol Multiplexing or by the assigned protocol. This bit is 0 when no protocol is assigned to Channel 3 (i.e., Channel 3 Assignment field is 0h).	RO
25	**PMUX Channel 3 Disabled: Link Width** – If Set, Channel 3 is disabled because the current Link Width is not supported by the assigned protocol. This bit is 0 when no protocol is assigned to Channel 3 (i.e., Channel 3 Assignment field is 0h).	RO
26	**PMUX Channel 3 Disabled: Protocol Specific** – If Set, Channel 3 is disabled for protocol specific reasons. This bit is 0 when no protocol is assigned to Channel 3 (i.e., Channel 3 Assignment field is 0h).	RO
32:27	**Reserved**	RsvdZ

G.5.5. PMUX Protocol Array (Offsets 10h Through 48h)

The PMUX Protocol Array consists of up to 63 entries. The size of the PMUX Protocol Array is indicated by the PMUX Protocol Array Size field (see Section G.5.2).

Figure G-12 details the allocation of fields in each PMUX Protocol Array entry. Table G-10 provides the respective bit definitions.

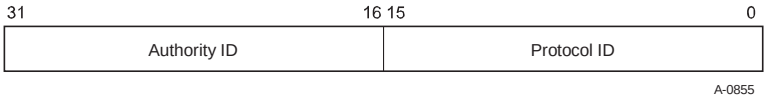


Figure G-12: PMUX Protocol Array Entry

Table G-10: PMUX Protocol Array Entry

Bit Location	Register Description	Attributes
15:0	**Protocol ID** – In conjunction with Authority ID designates a specific protocol and the mechanism by which that protocol is mapped onto Protocol Multiplexing.	RO
31:16	**Authority ID** – Designates the authority controlling the values used in the Protocol ID field. The Authority ID field contains a Vendor ID as assigned by the PCI-SIG.	RO

The value 0000 0000h indicates an unimplemented PMUX Protocol Array entry. The PMUX Protocol Array is indexed starting at 1.

PMUX Channel *n* is enabled and configured to support the protocol associated with PMUX Protocol Array entry at index *m* when the PMUX Channel *n* Assignment field contains the value *m* (see Section G.5.3).

Entries in the PMUX Protocol Array with the Authority ID value 1 (0001h) represent protocols that are defined by the PCI-SIG.

Duplicate Entries in the PMUX Protocol Array may be used to represent multiple instances of a particular protocol. This permits software control of the mapping between PMUX Channel ID and a specific instance of a protocol.



IMPLEMENTATION NOTE

Multiple Protocol Instances

A Link may have a single PMUX Protocol assigned to multiple PMUX Channels. Each PMUX Channel is assigned to a different instance of the protocol. Each instance of a protocol corresponds to an entry in the PMUX Protocol Array.

Consider a Port that supports two instances of protocol X. Two entries in the PMUX Protocol Array would indicate protocol X (indexes A and B for example). To assign instance A to PMUX Channel 0 and instance B to PMUX Channel 2, place the value A in the PMUX Channel 0 Assignment field and the value B in the PMUX Channel 2 Assignment field.



H. M-PCIe Timing Diagrams

This Section shows a few examples of Timing Diagrams to illustration the process from Initialization to L0 states, and the state transitions thereafter. These examples are informative only. In case of conflicts, please refer to the text specifications.

Comment [VS6]: Appendix H added per ECN: M-PCIe

H.1. Init to L0

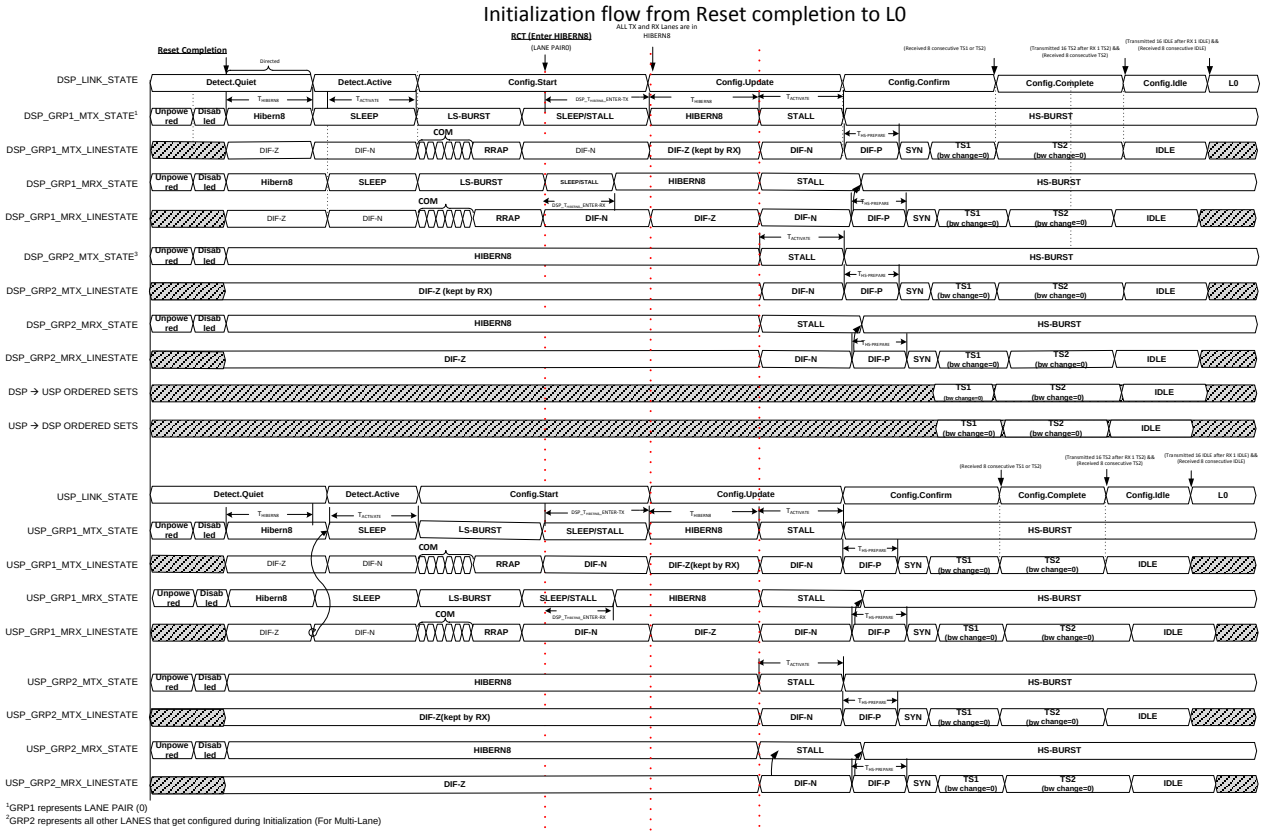


Figure H-1: Reset Completion to L0 Timing Diagram

H.2. L0 with Transmitter in STALL

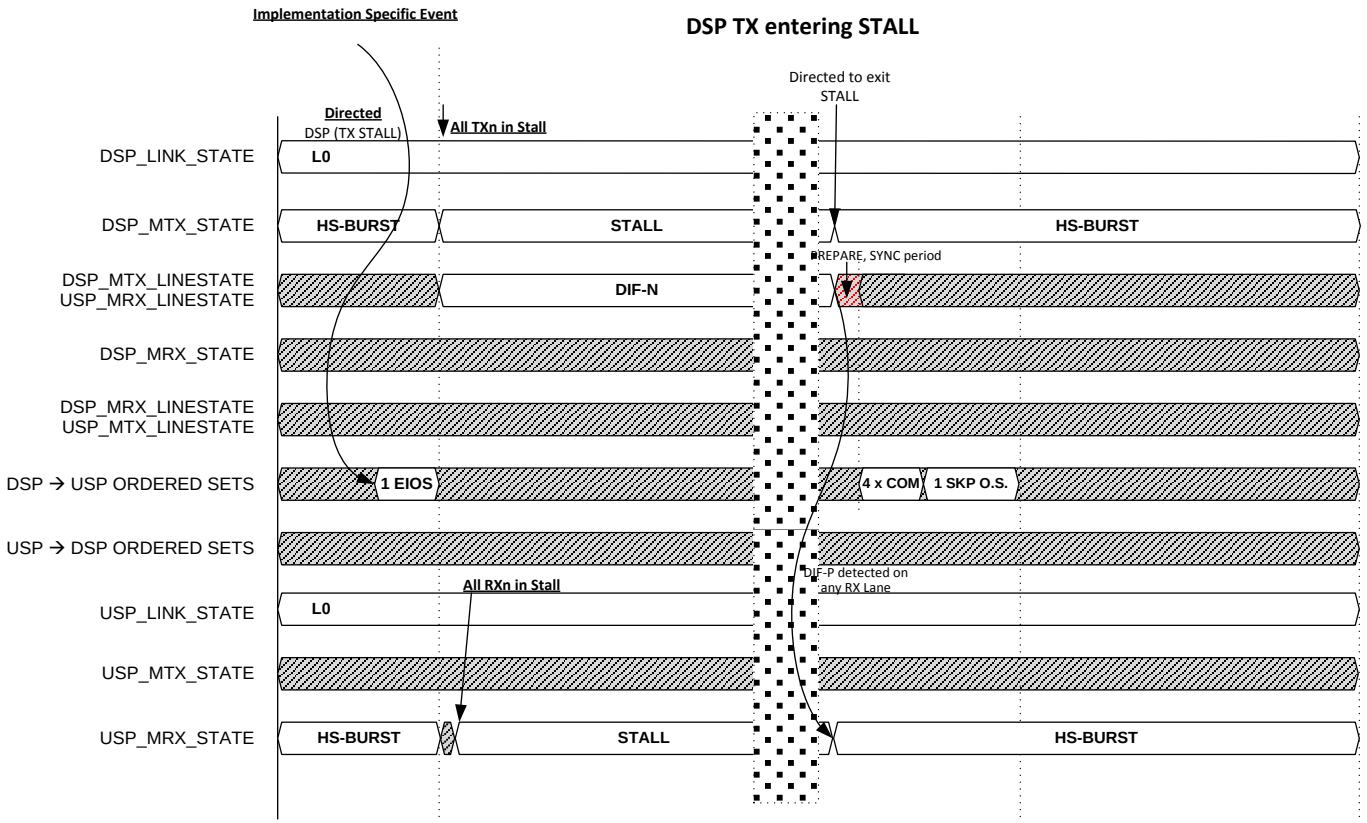


Figure H-2: L0 to Transmitter in STALL Timing Diagram

H.3. L0 to L1

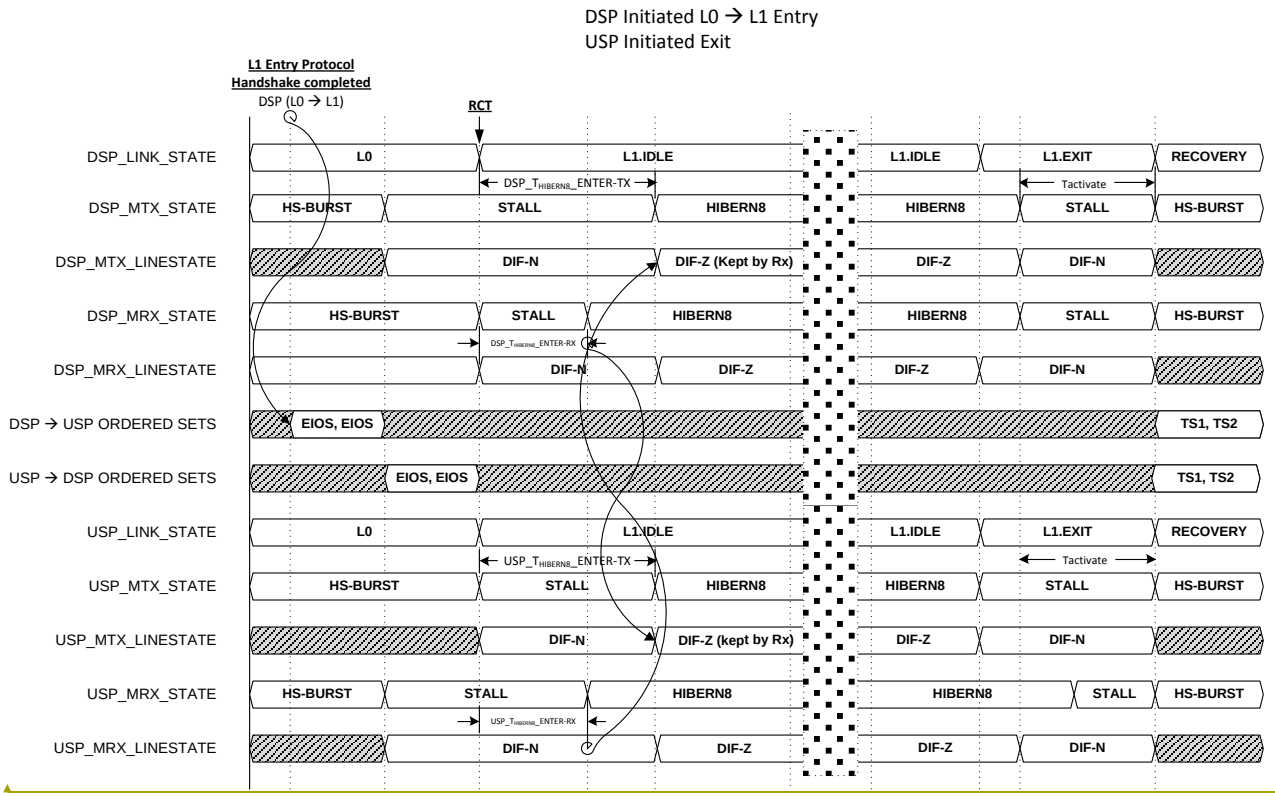


Figure H-3: DSP Initiated L0 to L1 Entry Timing Diagram

Formatted: Font color: Red

H.4. Downstream Port Initiated LINK Bandwidth Change

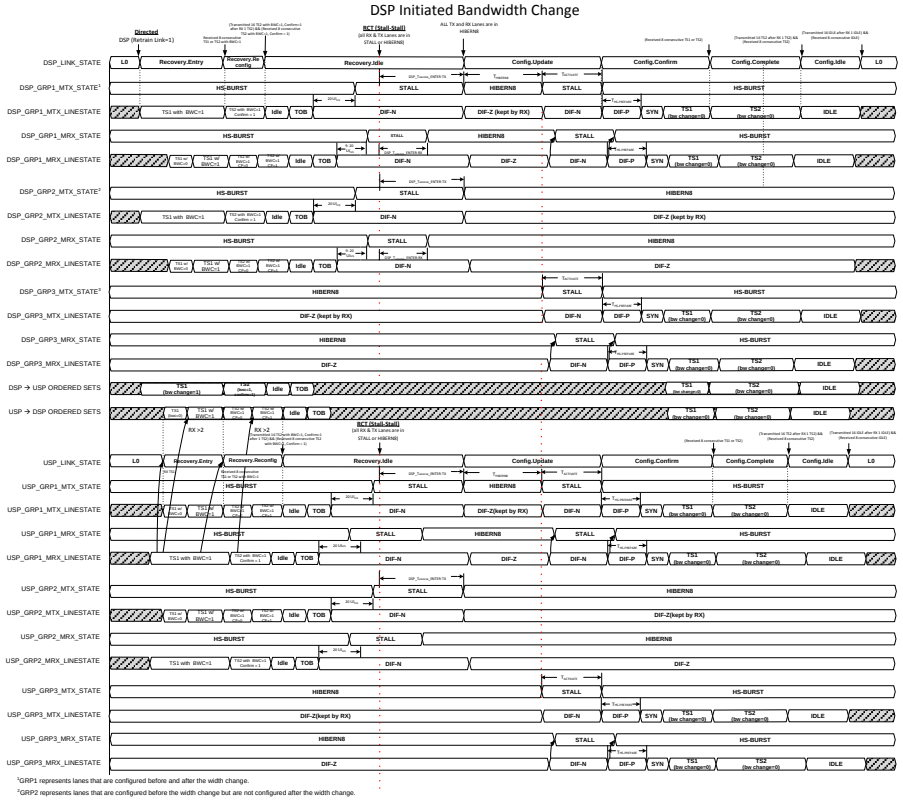
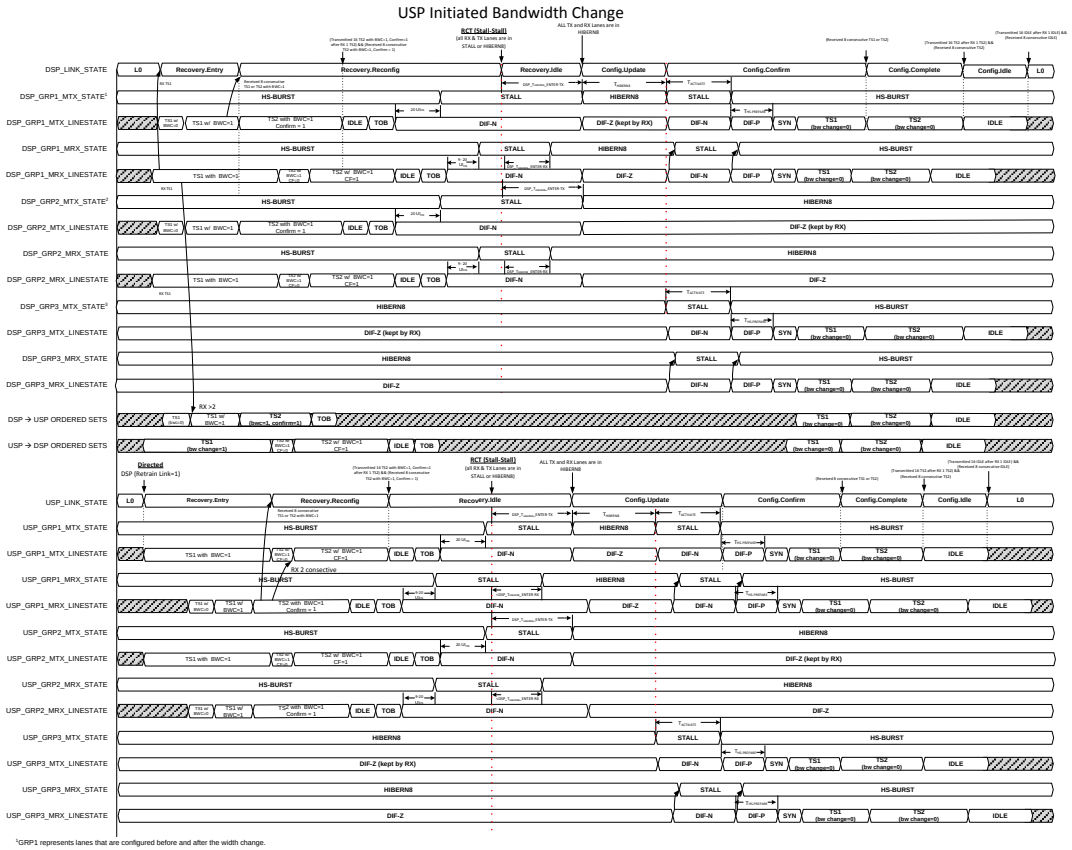


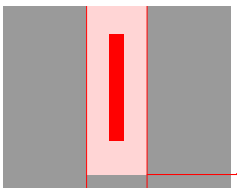
Figure H-4: Downstream Port Initiated LINK Bandwidth Change Timing Diagram

H.5. Upstream Port Initiated LINK Bandwidth Change



[1]GRP1 represents lanes that are configured before and after the width change.
[2]GRP2 represents lanes that are configured before the width change but are not configured after the width change.
[3]GRP3 represents lanes that are not configured before the width change but are configured after the width change.

Figure H-5: Upstream Port Initiated LINK Bandwidth Change Timing Diagram



Comment [VS7]: Appendix I added per ECN: M-PCIe

I. M-PCIe Compliance Patterns

Comment [VS8]: Title Changed from CRPAT: Editorial per Dave Harriman/Steve Glaser

This appendix defines continuous and burst mode CRPAT for TX compliance mode. While the continuous mode CRPAT is mandatory the support for a burst mode CRPAT is optional and recommended only. Burst mode CRPAT is used by the MIPI M-PHY CTS v1.0.

I.1. RPAT

The basic random data pattern RPAT used for CRPAT is based on the pattern originally defined by Fibre Channel and is given in Table I-1. Note that only the 8B10B encoded symbols are transmitted on the M-PHY lanes. Running disparity is not shown in Table I-1 but must be maintained at all times during TX compliance mode. The initial running disparity for TX compliance mode is not defined and left to the implementation.

Table I-1: RPAT Definition

#	8B Data Hex	8B10B Code Symbol			
1	be	D	30	.	5
2	d7	D	23	.	6
3	23	D	3	.	1
4	47	D	7	.	2
5	6b	D	11	.	3
6	8f	D	15	.	4
7	b3	D	19	.	5
8	14	D	20	.	0
9	5e	D	30	.	2
10	fb	D	27	.	7
11	35	D	21	.	1
12	59	D	25	.	2

I.2. RPAT Variation by Lane

When testing multiple lanes simultaneously it is intended to capture interference impact between lanes. Therefore the test pattern must accommodate for the multi-lane test case. Thus RPAT as given in Table I-1 is rotated by 2 Bytes per Lane. Because of the length of 12 Bytes every sixth lane uses the same RPAT. Table I-2 lists the 6 different versions of RPAT and shows how these are being used on different lanes.

Comment [VS9]: Mpcie_28August2014 Errata C4

Table I-2: RPAT shifted by 2 Bytes per Lane

#	Lane 0 8B Data Hex	Lane 1 8B Data Hex	Lane 2 8B Data Hex	Lane 3 8B Data Hex	Lane 4 8B Data Hex	Lane 5 8B Data Hex	Lane 6 8B Data Hex	…
1	be	23	6b	b3	5e	35	be	…
2	d7	47	8f	14	fb	59	d7	…
3	23	6b	b3	5e	35	be	23	…
4	47	8f	14	fb	59	d7	47	…
5	6b	b3	5e	35	be	23	6b	…
6	8f	14	fb	59	d7	47	8f	…
7	b3	5e	35	be	23	6b	b3	…
8	14	fb	59	d7	47	8f	14	…
9	5e	35	be	23	6b	b3	5e	…
10	fb	59	d7	47	8f	14	fb	…
11	35	be	23	6b	b3	5e	35	…
12	59	d7	47	8f	14	fb	59	…

I.3. Continuous Mode CRPAT

The continuous mode CRPAT repeatedly sends chunks of 38 repetitions of RPAT preceded by a Marker 0 on a given lane. While RPAT starts and ends with the same running disparity the Marker 0 changes it and thus the encoding for RPAT changes every block of 38 RPAT iterations.

Figure I-1 illustrates the continuous mode CRPAT for Lane 0 while SAVE mode signaling, SYNC phase and TAIL-OF-BURST are not detailed. When the test programmer directs the DUT to end the test sequence the current 38 repetitions of RPAT are completed and a Marker 2 appended to indicate the end of the test phase before the TAIL-OF-BURST.

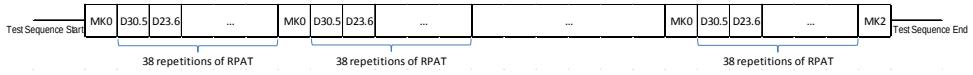


Figure I-1: Continuous Mode CRPAT (Lane 0 shown)

The insertion of Fillers is not allowed when the continuous mode CRPAT is being transmitted.

I.4. Burst Mode CRPAT

In burst mode CRPAT two chunks of 38 repetitions of RPAT both preceded by a Marker 0 are sent in one burst. As RPAT starts and ends with the same running disparity the Marker 0 between the first and second block of 38 RPATs ensures that both encodings are being transmitted. After the second block of 38 iterations of RPAT the burst is ended using the correct TAIL-OF-BURST mechanism and the TX enters the SAVE mode. ~~The duration of the SAVE mode should be kept at the minimum possible time and then the next CRPAT burst is being transmitted.~~ The duration of the SAVE mode must be kept as short as possible between transmitted burst mode CRPAT sequences.

Comment [VS10]: Mpcie_28August2014 Errata C4

Figure I-2 shows the burst mode CRPAT for Lane 0 without detailing TAIL-OF-BURST, SAVE mode signaling and SYNC phase. When the test programmer directs the DUT to end the test sequence the current burst is completed and a Marker 2 appended before the TAIL-OF-BURST to signal the end of the test phase. The insertion of Fillers is not allowed at any time when the burst mode CRPAT is being transmitted.

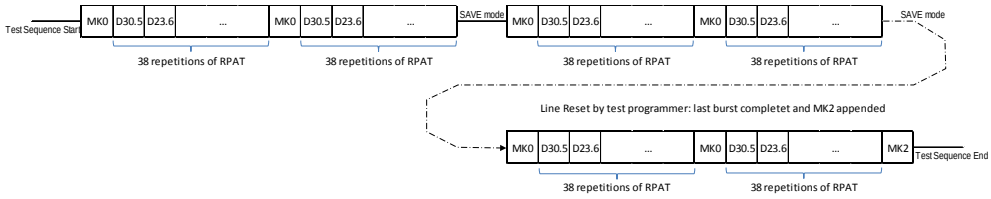


Figure I-2: Burst Mode CRPAT (Lane 0 shown)

Acknowledgements

The following persons were instrumental in the development of the PCI Express Base Specification:[117]

Chamath Abhayagunawardhana	Intel Corporation
Ryan Abraham	Oracle Corporation
Shiva Aditham	Intel Corporation
Shay Aisman	Mellanox Technologies, Ltd
Jasmin Ajanovic	Intel Corporation
Katsutoshi Akagi	NEC Corporation
Michael W. Altmann	Intel Corporation
Taha Amiralli	Advanced Micro Devices, Inc.
Sujith Arramreddy	ServerWorks, Inc.
Yuval Avnon	Marvell Semiconductor, Inc.
Jay Avula	ServerWorks, Inc.
Jasper Balraj	Intel Corporation
Nat Barbiero	Advanced Micro Devices, Inc.
Phillip Barnes	Hewlett-Packard Company
Suparna Behera	LSI Logic Corporation
Joseph A. Bennett	Intel Corporation
Richard Bell	Advanced Micro Devices, Inc.
Stuart Berke	Hewlett-Packard Company
Harish Bharadwaj	LSI Logic Corporation
Ajay V. Bhatt	Intel Corporation
Gaurav Bhide	Advanced Micro Devices, Inc.
Harmeet Bhugra	IDT Corporation
Bill Bissonette	Intel Corporation
Cass Blodget	Intel Corporation
Jeffrey D. Bloom	Intel Corporation
Naveen Bohra	Intel Corporation
Austin Bolen	Dell Inc.
Jerry Bolen	Intel Corporation
Michele Bologna	Synopsys, Inc.
Hicham Bouzekri	ST-Ericsson
Gavin Bowlby	QLogic Corporation
Suri Brahmaroutu	Dell Inc.
Dave Brown	Integrated Device Technology

[117] Company affiliation listed is at the time of specification contributions.

Tory Brown	Intel Corporation
Mike Brownell	Intel Corporation
Bala Cadambi	Intel Corporation
John Calvin	Tektronix, Inc.
Jun Cao	Marvell Semiconductor, Inc.
Gord Caruk	Advanced Micro Devices, Inc.
James Chapple	Intel Corporation
Kabitha Chaturvedula	LSI Logic Corporation
Yogesh Chaudhary	Mentor Graphics
Santanu Chaudhuri	Intel Corporation
Rajinder Cheema	LSI Logic Corporation
Albert Chen	LSI Logic Corporation
Chih-Cheh Chen	Intel Corporation
Qunwei Chen	Advanced Micro Devices, Inc.
Tony Chen	QLogic Corporation
Jonathan Cheung	Advanced Micro Devices, Inc.
Yorick Cho	Advanced Micro Devices, Inc.
Dickson Chow	Advanced Micro Devices, Inc.
Gene Chui	IDT Corporation
Shawn Clayton	Emulex Corporation
Mark Clements	IBM Corporation
Debra Cohen	Intel Corporation
Eric Combs	LSI Logic Corporation
Brad Congdon	Intel Corporation
Mike Converse	IDT Corporation
Justin Coppin	Hewlett-Packard Company
Joe Cowan	Hewlett-Packard Company
Carrie Cox	IBM Corporation
H. Clay Cranford	IBM Corporation
Kenneth C. Creta	Intel Corporation
Pamela Cristo	Emulex Corporation
Sanjay Dabral	Intel Corporation
Eric Dahlen	Intel Corporation
Tugrul Daim	Intel Corporation
Ron Dammann	Intel Corporation
Sumit Das	Texas Instruments Incorporated
Debendra Das Sharma	Intel Corporation
Nicole Daugherty	Intel Corporation
Glen Dearth	Advanced Micro Devices, Inc.
Fei Deng	Intel Corporation
Dan DeSetto	Intel Corporation

Karishma Dhruv	LSI Logic Corporation
Gary Dick	Cadence Design Systems, Inc
Bob Divivier	IDT Corporation
Ken Drottar	Intel Corporation
Dave Dunning	Intel Corporation
Rick Eads	Agilent Technologies, Inc.
Gregory L. Ebert	Intel Corporation
Yaron Elboim	Intel Corporation
Bassam Elkhoury	Intel Corporation
Salem Emara	Advanced Micro Devices, Inc.
Mike Engbretson	Tektronix, Inc.
Ohad Falik	Intel Corporation
David Fair	Intel Corporation
Blaise Fanning	Intel Corporation
John Feehrer	Oracle Corporation
Wes Ficken	IBM Corporation
Joshua Filliater	LSI Logic Corporation
Michael Fleischer-Reumann	Agilent Technologies, Inc.
Pelle Fornberg	Intel Corporation
Jim Foster	IBM Corporation
Mike Foxcroft	Advanced Micro Devices, Inc.
Daniel S. Froelich	Intel Corporation
SivaPrasad Gadey	Intel Corporation
Eric Geisler	Intel Corporation
Gordon Getty	Agilent Technologies, Inc.
Gregory Geyfman	Intel Corporation
Stefano Giaconi	Intel Corporation
Steve Glaser	nVidia Corporation ~~Advanced Micro Devices, Inc.~~
Marc A. Goldschmidt	Intel Corporation
Alan Goodrum	Hewlett-Packard Company
Robert Gough	Intel Corporation
Ilya Granovsky	IBM Corporation
Brien Gray	Intel Corporation
Richard Greene	Intel Corporation
Stephen Greenwood	ATI Technologies Inc.
Michael J. Greger	Intel Corporation
Buck Gremel	Intel Corporation
George Gruber	Qualcomm
Vijay Gudur	LSI Logic Corporation
Dale Gulick	Advanced Micro Devices, Inc.
Liping Guo	Marvell Semiconductor, Inc.

Mickey Gutman	Intel Corporation
Clifford D. Hall	Intel Corporation
Stephen H. Hall	Intel Corporation
Joseph Hamel	IBM Corporation
Ken Haren	Intel Corporation
David Harriman	Intel Corporation
Hiromitsu Hashimoto	NEC Corporation
George R. Hayek	Intel Corporation
Wenmu He	Texas Instruments Incorporated
Matt Hendrick	Intel Corporation
Hamish Hendry	Cadence Design Systems, Inc
Michael Herz	Blackberry
Bent Hessen-Schmidt	Tektronix, Inc.
Hanh Hoang	Intel Corporation
Joe Hock	Intel Corporation
Michael Hopgood	nVidia Corporation
Dorcas Hsia	nVidia Corporation
James Huang	Advanced Micro Devices, Inc.
Robert Huang	nVidia Corporation
Kamlesh Hujwant	Advanced Micro Devices, Inc.
Mark Hummel	Advanced Micro Devices, Inc.
Mikal Hunsaker	Intel Corporation
Yasser Ibrahim	Microsoft Corporation
Franko Itay	Intel Corporation
Carl Jackson	Hewlett-Packard Company
David R. Jackson	Intel Corporation
Praveen Jain	nVidia Corporation
Martin James	Cadence Design Systems, Inc
Duane January	Intel Corporation
James E. Jaussi	Intel Corporation
Mike Jenkins	LSI Logic Corporation
Peter Jenkins	IBM Corporation
Hong Jiang	Intel Corporation
Viek Joshi	Intel Corporation
Dave Kaffine	Oracle Corporation
David Kahn	Oracle Corporation
Ravi Kammaje	LSI Logic Corporation
Girish Karanam	LSI Logic Corporation
Chad Kendall	Broadcom Corporation
Mukund Kharti	Dell Computer Corporation
David Kimble	Texas Instruments Incorporated

Mohammad Kolbehdari	Intel Corporation
Abhimanyu Kolla	Intel Corporation
Ganesh Kondapuram	Intel Corporation
Michael Krause	Hewlett-Packard Company
Sreenivas Krishnan	nVidia Corporation
Kumaran Krishnasamy	Broadcom Corporation
Steven K. Krooswyk	Intel Corporation
Manjari Kulkarni	Intel Corporation
Akhilesh Kumar	Intel Corporation
Mohan J. Kumar	Intel Corporation
Richard Kunze	Intel Corporation
Hugh Kurth	Oracle Corporation
Seh Kwa	Intel Corporation
Sunny Lam	Intel Corporation
Dave Landsman	Sandisk
Brian Langendorf	Intel Corporation
Raymond R. Law	Intel Corporation
Dror Lazar	Intel Corporation
Brian L'Ecuyer	Agilent Technologies, Inc.
Beomtek Lee	Intel Corporation
Clifford D. Lee	Intel Corporation
David M. Lee	Intel Corporation
Edward Lee	Advanced Micro Devices, Inc.
Moshe Leibowitz	IBM Corporation
Tony L. Lewis	Intel Corporation
Mike Li	Wavecrest Corporation
Paul Li	Pericom Semiconductor Corporation
Stephen Li	Texas Instruments Incorporated
Andrew K. Lillie	Intel Corporation
Wendy Liu	Advanced Micro Devices, Inc.
Jeff Lukanc	IDT Corporation
Jeffrey Lu	IBM Corporation
Betty Luk	Advanced Micro Devices, Inc.
Ngoc Luu	Advanced Micro Devices, Inc.
Kenneth Ma	Broadcom Corporation
Stephen Ma	Advanced Micro Devices, Inc.
Zorik Machulsky	IBM Corporation
Kevin Main	Texas Instruments Incorporated
Steven Makow	IBM Corporation
Tony Mangefeste	Microsoft Corporation
Nilange Manisha	Intel Corporation
Steve Manning	Advanced Micro Devices, Inc.

Gold Mao	VIA Technologies, Inc.
Jarek Marczewski	Advanced Micro Devices, Inc.
Mark Marlett	LSI Logic Corporation
Alberto Martinez	Intel Corporation
Andrew Martwick	Intel Corporation
Paul Mattos	IBM Corporation
Robert A. Mayer	Intel Corporation
Mehdi M. Mechaik	nVidia Corporation N
Pranav H. Mehta	Intel Corporation
Vishal Mehta	nVidia Corporation
Richard Mellitz	Intel Corporation
Cindy Merkin	Dell Computer Corporation
Dennis Miller	Intel Corporation
Jason Miller	Oracle Corporation
Robert J. Miller	Intel Corporation
Michael Mirmak	Intel Corporation
Suneel Mitbander	Intel Corporation
Daniel Moertl	IBM Corporation
Lee Mohrmann	National Instruments Corporation
Puga Nathal Moises	Intel Corporation
Wayne Moore	Intel Corporation
Douglas R. Moran	Intel Corporation
Terry Morris	Hewlett-Packard Company
Jeff C. Morriss	Intel Corporation
Linna Mu	Avago Technologies
Sridhar Muthrasanallur	Intel Corporation
Suresh Babu M. V.	LSI Logic Corporation
Gautam V. Naik	LSI Logic Corporation
Mohan K. Nair	Intel Corporation
Mukund Narasimhan	Intel Corporation
Alon Naveh	Intel Corporation
Ramin Neshati	Intel Corporation
Surena Neshvad	Intel Corporation
Andy Ng	IDT Corporation
Manisha Nilange	Intel Corporation
Hajime Nozaki	NEC Corporation
Kugao Ohuchi	NEC Corporation
Olufemi Oluwafemi	Intel Corporation
Takuya Omura	Synopsys, Inc.
Peter Onufryk	Integrated Device Technology
Mike Osborn	Advanced Micro Devices, Inc.

Randy Ott	Intel Corporation
Jonathan Owen	Advanced Micro Devices, Inc.
Ali Oztaskin	Intel Corporation
David Pabisz	Oracle Corporation
Shreeram Palghat	Intel Corporation
Jim Panian	Qualcomm
Marc Pegolotti	ServerWorks, Inc.
Henry Peng	Intel Corporation
Akshay Pethe	Intel Corporation
Lu-vong Phan	Intel Corporation
Tony Pierce	Microsoft Corporation
Edmund Poh	Molex, Inc.
Harshit Poladia	Intel Corporation
Jim Prijic	Intel Corporation
Dave Puffer	Intel Corporation
Duane Quiet	Intel Corporation
Jeffrey D. Rabe	Intel Corporation
Andy Raffman	Microsoft Corporation
Kianoush Rahbar	Intel Corporation
Guru Rajamani	Intel Corporation
Ramesh Raman	LSI Logic Corporation
Adee Ran	Intel Corporation
Todd Rasmus	IBM Corporation
Ramakrishna Reddy	LSI Logic Corporation
Jack Regula	PLX Technology, Inc.
Dick Reohr	Intel Corporation
Curtis Ridgeway	LSI Logic Corporation
Dwight Riley	Hewlett-Packard Company
Yoav Rozenbert	Mellanox Technologies, Ltd
Chris Runhaar	nVidia Corporation
Rajanataraj S.	LSI Logic Corporation
Devang Sachdev	nVidia Corporation
Gene Saghi	LSI Logic Corporation
Bill Sauber	Dell Computer Corporation
Mike Schaecher	Oracle Corporation
Joe Schaefer	Intel Corporation
Michael Scheid	Broadcom Corporation
Daren Schmidt	Intel Corporation
Mark Schmisseur	Intel Corporation
Richard Schober	nVidia Corporation
Zale Schoenborn	Intel Corporation
Rick Schuckle	Dell Computer Corporation

Richard Schumacher	Hewlett-Packard Company
Jeremiah Schwartz	Intel Corporation
Tudor Secasiu	Intel Corporation
Kazunori Seki	~~Inventure~~Synopsys, Inc.
Oren Sela	Mellanox Technologies, Ltd
Kevin Senohrabek	Advanced Micro Devices, Inc.
Kalev Sepp	Tektronix, Inc.
Prashant Sethi	Intel Corporation
Ankur Shah	Intel Corporation
Vasudevan Shanmugasundaram	Intel Corporation
Wesley Shao	Oracle Corporation
Prateek Sharma	LSI Logic Corporation
Charlie Shaver	Hewlett-Packard Company
Robert Sheldon	Oracle Corporation
John Sheplock	IBM Corporation
Milton Shih	Oracle Corporation
Mark Shillingburg	Agilent Technologies
Oren Shirak	Mellanox Technologies, Ltd
Sarvesh Shrivastava	Qualcomm
Bill Simms	nVidia Corporation
Dan Slocombe	Cadence Design Systems, Inc
Brian Small	Northwest Logic, Inc
George Smith	Microsoft Corporation
Shamnad SN	LSI Logic Corporation
Rick Sodke	PMC-Sierra
Gary Solomon	Intel Corporation
Richard Solomon	Synopsys, Inc~~LSI Logic Corporation~~
Gao Song	IDT Corporation
Walter Soto	Broadcom Corporation
Fulvio Spagna	Intel Corporation
Jason Squire	Molex, Inc.
Patrick Stabile	Oracle Corporation
Stan Stanski	IBM Corporation
Ron Swartz	Intel Corporation
Tim Symons	PMC-Sierra
Miki Takahashi	NEC Corporation
Gerry Talbot	Advanced Micro Devices, Inc.
Anthony Tam	Advanced Micro Devices, Inc.
Wanru Tao	Oracle Corporation
Mark Taylor	nVidia Corporation
Matthew Tedone	LSI Logic Corporation

Grigori Temkine	Advanced Micro Devices, Inc.
Peter Teng	NEC Corporation
Bruce A. Tennant	Intel Corporation
Larry Tesdall	QLogic Corporation
Andrew Thornton	Microsoft Corporation
Mike Tobin	Intel Corporation
Tan V. Tran	Intel Corporation
Alok Tripathi	Intel Corporation
William Tsu	nVidia Corporation
Chris Van Beek	Intel Corporation
Arie van der Hoeven	Microsoft Corporation
Andrew Vargas	Intel Corporation
Robertson Velez	ATI Technologies Inc
Archana Vasudevan	LSI Logic Corporation
Kiran Velicheti	Intel Corporation
Balaji Vembu	Intel Corporation
Gary Verdun	Dell Computer Corporation
Sushil Verghese	Broadcom Corporation
Divya Vijayaraghavan	Altera Corporation
Ravindra Viswanath	LSI Logic Corporation
Pete D. Vogt	Intel Corporation
Andrew Volk	Intel Corporation
Mahesh Wagh	Intel Corporation
Clint Walker	Intel Corporation
Davis Walker	Microsoft Corporation
Hui Wang	IDT Corporation
Kai A. Wang	Intel Corporation
Leo Warmuth	NXP Semiconductors
Dan Wartski	Intel Corporation
Neil Webb	Cadence Design Systems, Inc
Eric Wehage	Intel Corporation
Dong Wei	Hewlett-Packard Company
Ron Weimer	QLogic Corporation
Amir Wiener	Intel Corporation
Marc Wells	Intel Corporation
Rob Wessel	Hewlett-Packard Company
Stephen F. Whalley	Intel Corporation
Bryan White	Intel Corporation
Paul Whittemore	Oracle Corporation
Theodore L. Willke	Intel Corporation
Dawn Wood	Intel Corporation
John Wood	Emulex Corporation

David Woodral	QLogic Corporation
David Wooten	Microsoft Corporation
Zhi Wong	Altera Corporation
David Wormus	LSI Logic Corporation
David Wu	Broadcom Corporation
William Wu	Broadcom Corporation
Zuoguo Wu	Intel Corporation
Liu Xin	IDT Corporation
Dan Yaklin	Texas Instruments Incorporated
Howard Yan	Intel Corporation
Jane Yan	Oracle Corporation
Al Yanes	IBM Corporation
Gin Yee	Advanced Micro Devices, Inc.
Ahmed Younis	Xilinx, Inc.
Chi-Ho Yue	nVidia Corporation
Wayne Yun	Advanced Micro Devices, Inc.
Dave Zenz	Dell Computer Corporation
Shubing Zhai	IDT Corporation
Bo Zhang	Intel Corporation
Guoqing Zhang	Cadence Design Systems, Inc